NIST Special Publication 800-85A-2

PIV Card Application and Middleware Interface Test Guidelines (SP 800-73-3 compliance)

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

Ramaswamy Chandramouli
Hildegard Ferraiolo
Ketan Mehta

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

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

July 2010

U.S. Department of Commerce
Gary Locke, Secretary

National Institute of Standards and
Patrick D. Gallagher, 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 analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

Acknowledgements

The authors (Ramaswamy Chandramouli, Hildegard Ferraiolo of NIST, and Ketan Mehta of Booz Allen Hamilton) wish to thank their colleagues who reviewed drafts of this document and contributed to its development. The authors also gratefully acknowledge and appreciate the many contributions from the public and private sectors whose thoughtful and constructive comments improved the quality and accuracy of this publication.

Table of Contents

List of Figures

List of Tables

1. Introduction

The Personal Identity Verification (PIV) of Federal Employees and Contractors, Federal Information Processing Standard 201 (FIPS 201) [1] was developed to establish standards for identity credentials. FIPS 201 sets the minimum requirements for a federal personal identification system that meets the control and security objectives of Homeland Security Presidential Directive (HSPD) 12 [2]. FIPS 201 also gives the technical specifications of components and processes required for the interoperability of PIV Cards[1] with the access control and PIV card management systems throughout the Federal Government. FIPS 201 is accompanied by three documents:

- National Institute of Standards and Technology Special Publication 800-73-3 (NIST SP 800-73-3) [3] specifies interface requirements for retrieving and using the identity credentials from the PIV Card. It also defines a PIV data model, which details the structure and format of the information stored on the PIV Card.
- NIST SP 800-76-1 [4] contains technical specifications for biometric data mandated in FIPS 201.
- NIST SP 800-78-2 [5] specifies the cryptographic algorithms and key sizes for performing cryptographic operations on PIV data objects defined as part of the PIV data model.

This test guidance document specifies the test plan, processes, derived test requirements, and the detailed test assertions/conformance tests for testing the following PIV software components:

- PIV Middleware (implements PIV Client API)
- PIV Card Application.

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 security for sensitive unclassified information in Federal computer systems. This recommendation is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided A-130, Appendix III.

This recommendation has been prepared for use by federal agencies. It may be used by non-governmental organizations on a voluntary basis and is not subject to copyright. 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

[1] The term PIV Card in the context of this document refers to a smart card loaded with a PIV Card Application.

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

The objective of this document is to provide test requirements and test assertions that could be used to validate the compliance/conformance of two PIV components—*PIV Middleware* and PIV *Card Applications* with the specifications in NIST SP 800-73-3. Because NIST SP 800-73-3 specifications were developed for meeting interoperability goals of FIPS 201, the conformance tests in this document provide the assurance that the set of PIV Middleware and PIV Card Applications that have passed these tests are interoperable. This in turn facilitates marketing and procurement of FIPS 201-conformant products that meet the goals of HSPD-12.

1.3 Scope

This document provides guidelines for running conformance tests for the following three classes of specifications in NIST SP 800-73-3:

- End-Point PIV Data Objects Representation (Chapter 4, Part 1 of SP 800-73-3) and End-Point Data Types and Their Representation (Chapter 5, Part 1 of SP 800-73-3).
- End-Point PIV Card Application Card Command Interface (Part 2 of SP 800-73-3).
- End-Point PIV Client Application Programming Interface (Part 3 of SP 800-73-3).

The functions specified in the End-Point Client Application Programming Interface are to be supported by PIV Middleware. The commands specified in the End-Point PIV Card Application Card Command Interface are to be supported by PIV Card Application with appropriate security conditions for executing each command and for accessing/storing each of the data objects associated with the application. The overall design of the commands has to be based on the concepts outlined in End-Point Concepts and Constructs. The presence of mandatory data objects on the PIV card has to be verified. The data objects associated with PIV Card Application have to be tested for their accessibility and storage using the specified identifiers. Thus, the three classes of specifications listed above span the following two main PIV components: PIV Middleware and PIV Card Application. Hence the test suite provided in this document consists of the following two broad categories of tests:

- PIV Middleware tests
- PIV Card Application tests

The above tests are developed through the following two-step process:

- **Derived Test Requirements (DTR).** These are constructed from the ‘Shall’ statements in SP 800-73-3 specifications.
- **Test assertions.** These provide the tests that need to be performed to test each of the requirements under DTRs as well as tests with appropriate execution conditions for each of the commands in the interface to realize the associated return/response status codes specified in SP 800-73-3 Part 2.

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, card reader/biometric reader drivers, and specialized service provider software such as Cryptographic Service Provider (CSP) modules and Biometric Service Provider Modules. This document does not address nor provide conformance tests for SP 800-76 or FIPS 201.

1.4 Target Audience

This document is intended to:

- Enable developers of PIV Middleware and the PIV Card Applications to develop their software modules to be testable for interface requirements specified in SP 800-73-3.
- Enable developers of PIV Middleware and the PIV Card Applications to develop self-tests as part of the development effort.
- Enable testing laboratories authorized to perform conformance tests on PIV Middleware and the PIV Card Application to develop tests that cover the test suite provided in this document.

1.5 Document Overview

The document is organized as follows:

- Section 2 provides a conceptual software overview of a typical PIV system and introduces the PIV test components.
- Section 3 lists the various elements of the test suite under the two broad categories of tests (PIV Middleware tests and PIV Card Application tests) provided in this document.
- Section 4 provides an overview of the DTR construction process.
- Section 5 gives a brief description of the test assertion for each of the three specification classes covered by this document (refer to Section 1.3).
- Section 6 explains the documentation required from both the component owners and test labs for conducting the testing process.
- Section 7 details the acceptance criteria for each type of test.
- Section 8 explains the test compliance process and failure review.
- Appendix A includes DTRs based on specifications in SP 800-73-3.
- Appendix B includes client Application Programming Interface (API) test assertions.
- Appendix C includes PIV card command interface test assertions.
- Appendix D provides the format for reporting of test results by referencing the relevant sections from Appendices B and C that describe the corresponding test scenarios.

- Appendix E provides the requirements traceability matrix from DTRs to the test scenarios in Appendix B and Appendix C.
- Appendix F discusses implementation considerations for PIV Middleware.
- Appendix G describes the textual representations used in the document.
- Appendix H contains the list of documents used as references by this document.

2. System Overview

The conceptual architecture involving the PIV Middleware and PIV Card Application for which conformance tests are given in this document is shown in Figure 1. The conformance tests in this document apply to the areas highlighted with dashed lines in the Figure 1.

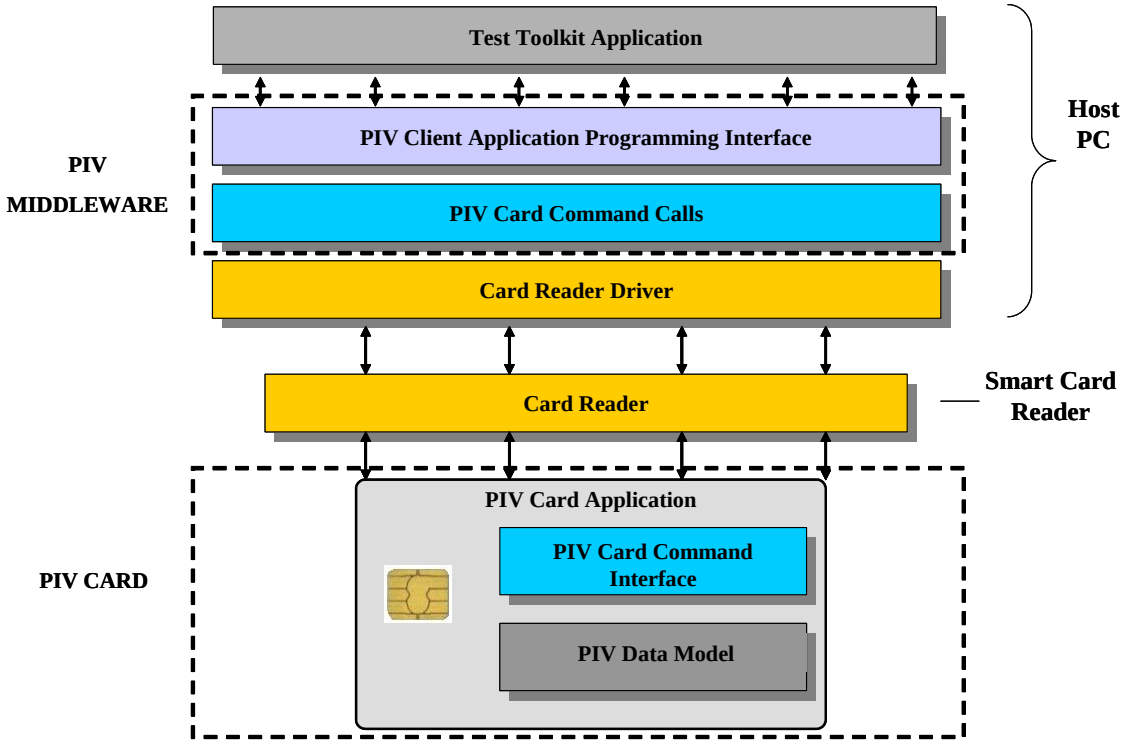


Figure 1: PIV Conformance Test Architecture

- PIV Middleware is a software application that is the interface between an agency's PIV implementation and the PIV Card Application. It allows the agency applications to remain independent of the underlying operating system platform. The PIV Middleware has the following two functions.

  1. It implements the functions in the PIV Client Application Programming Interface (Part 3 of SP 800-73-3).

  2. It generates the appropriate commands (also called Application Protocol Data Units or APDUs) for the PIV Card Command Interface (card edge interface - Part 2 of SP 800-73-3) and thus communicates with the PIV Card Application.

- The PIV Card Application resides on the card, implements the commands in the PIV Card Command Interface (Part 2 of SP 800-73-3), and provides access to objects of the PIV Data Model. The PIV Data model defines the logical use of the on-Card Application

space including the SP 800-73-3 Part 1 required data objects and data elements, along with the size and structure of each object.

2.1 Test Plan

The test plan identifies the tasks/artifacts required for testing the PIV Middleware and PIV Card Applications. These artifacts include the following: PIV Middleware and smart card populated with a PIV Card Application; the test toolkit (or test scripts), which implements the test assertions; and the various infrastructure devices needed to interface with the card, such as the card acceptance device (called the card reader). The components involved in the test plan and the elements of the test configuration for the two broad categories of tests presented in this document are discussed in the next two subsections.

2.2 Test Set-up

The test system consists of the following components:[2]

- A test toolkit application software that resides on a personal computer (PC)
- Smart card (SC) readers:
  - An ISO/IEC 7816 and PC/SC-compliant contact-based smart card reader and
  - An ISO/IEC 14443 and PC/SC-compliant contactless smart card reader

  or
  - A dual interface reader
- A mechanism to input personal identification number (PIN) (e.g. a PIN pad or a keyboard) that can be transmitted to the SC Reader.
- A set of test PIV Cards, loaded with PIV Card Application, with a contact interface that is compliant with ISO/IEC 7816 and a contactless interface that is compliant with ISO/IEC 14443 or a test PIV card emulator
- PIV Middleware application

These components will be used in different configurations based on the type of test being conducted in the test bed.

2.3 Test System Configuration

The test system shown in Figure 2 will be configured in both the PIV Middleware tests and the PIV Card Application tests to accommodate the different components to be tested, as explained in Section 3.

[2] Compliance of the readers and input devices with an external standard such as ISO/IEC 7816 is not addressed in this document.

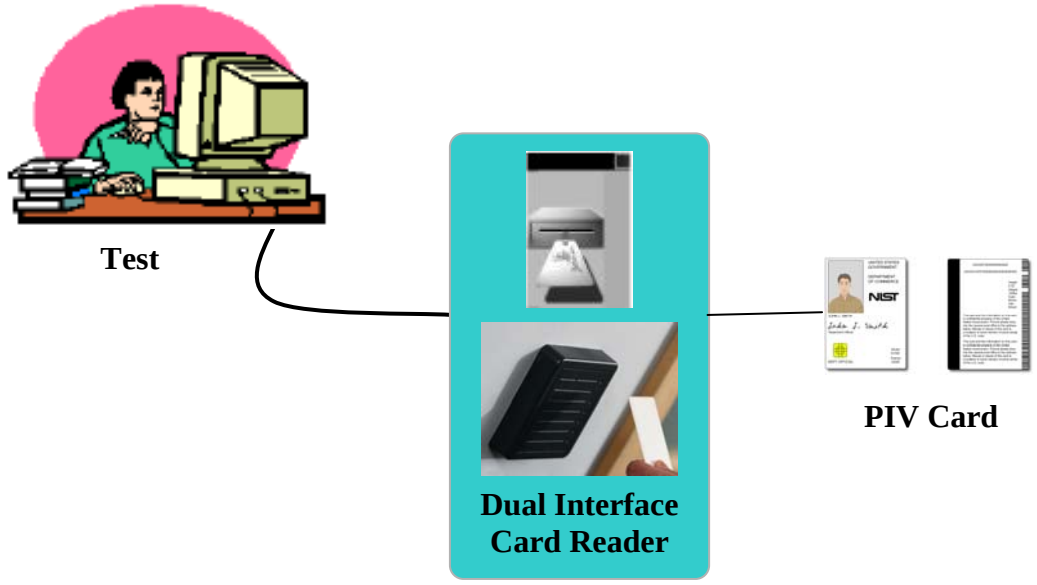


Figure 2: Test System Configuration

2.3.1 PIV Middleware Test Configuration

The middleware test configuration is used to test a vendor's middleware software application that implements the PIV client API and generates the appropriate commands in the PIV card command interface (refer to Table A-1 for mapping between the client API and card command interface). The middleware test configuration is depicted in Figure 3.

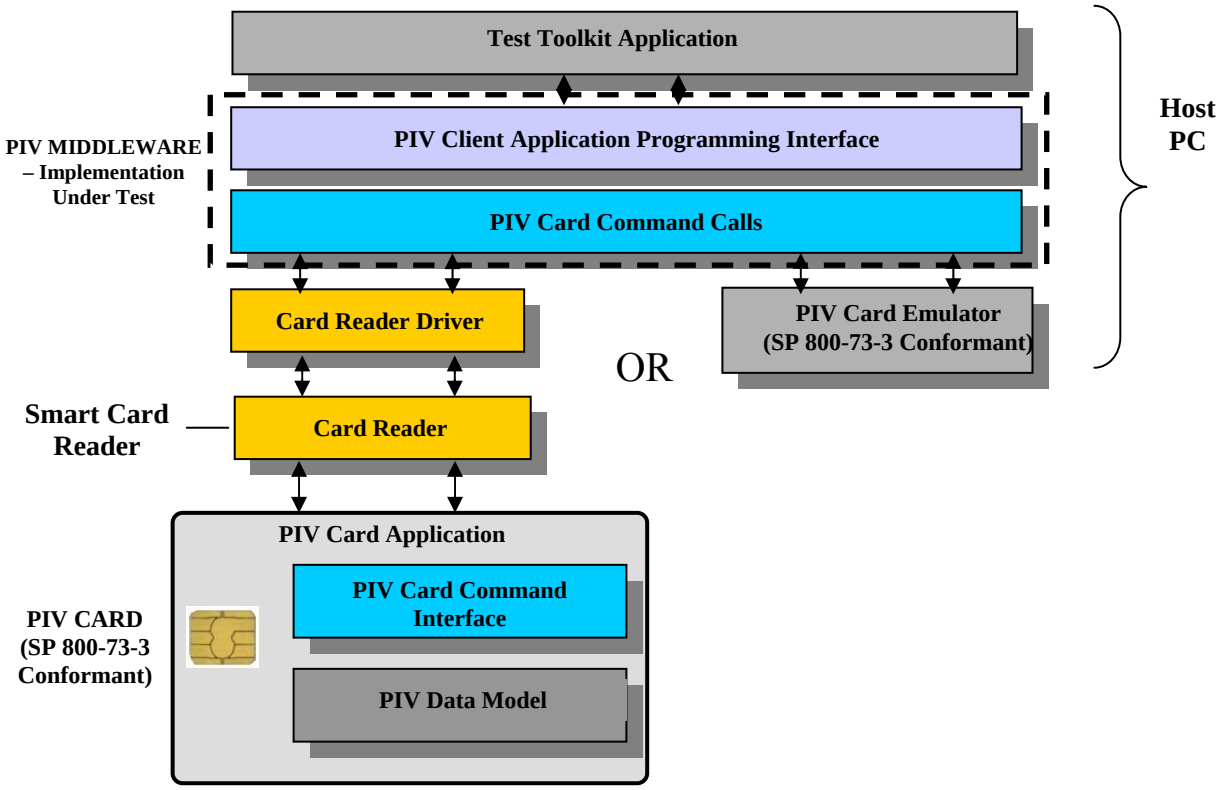


Figure 3: Middleware Test Configuration

The following list shows the test system components included in this configuration:

- A test toolkit application software
- Vendor provided PIV Middleware, which is the subject of this test (also called as Implementation Under Test or IUT) and

One of the following combinations:

- Contact and contactless smart card readers or a dual interface reader together with
- A PIN input mechanism together with
- A dual interface FIPS 201 conformant PIV Card loaded with "SP 800-73-3 Part 2 conformant PIV Card Application" (for definition refer to section 7.2)

OR

- A PIV card emulator that emulates the behavior of a PIV Card Application
- A printer for reporting and documenting the test results

The test toolkit application software resides on the Test Computer, and facilitates the execution and management of both test suites explained in Section 3. For the PIV Middleware Test, the test system (Figure 2) will be configured so that the vendor provided PIV Middleware under test is also installed on the Test Computer and interacts with the FIPS 201 conformant test cards via the card reader(s).

2.3.2 PIV Card Application Test Configuration

The card application test configuration is used to test any PIV Card Application through commands of the PIV card command interface defined in SP 800-73-3 Part 2. The following list shows the test system components included in this configuration:

- A test toolkit application software
- Contact and contactless smart card readers or a dual interface reader
- A PIN input mechanism
- A PIV Card loaded with PIV Card Application that supports contact and contactless interface and is the subject of this test. (also called Implementation Under Test or IUT)

For the PIV Card Application Test, the test system shown in Figure 2 will be configured such that the test toolkit application software directly interacts with the PIV Card under test via the card reader(s). The PIV Card Application Test configuration is depicted in Figure 4.

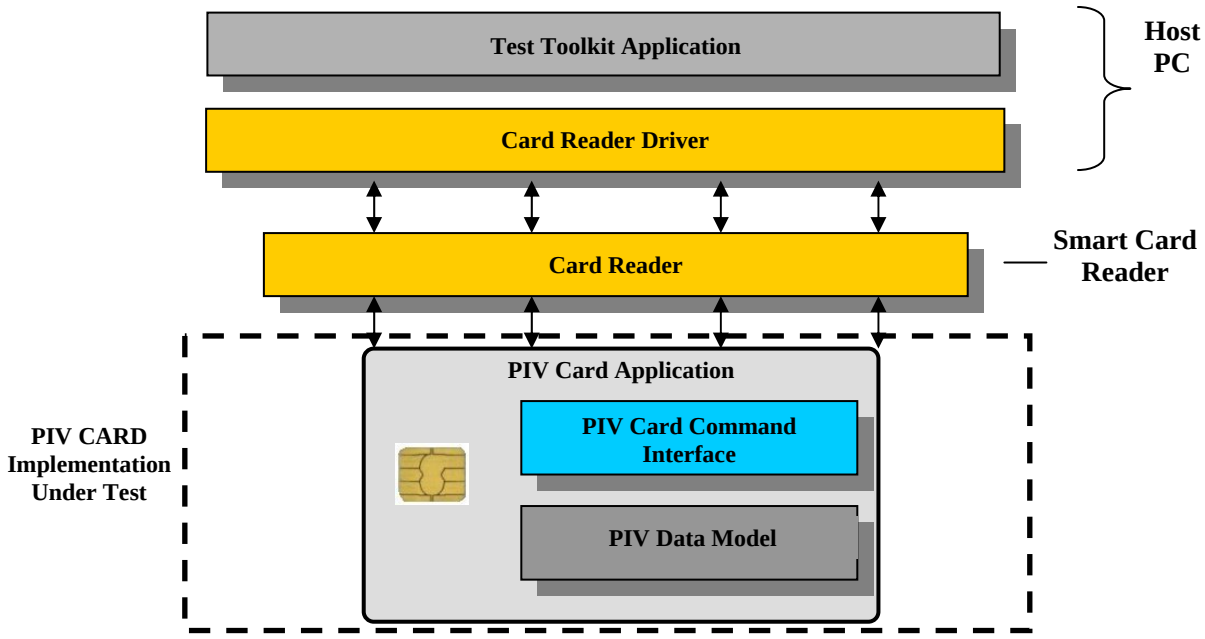


Figure 4: PIV Card Application Test Configuration

3. Test Suite Elements

Based on the conceptual software architecture shown in Figure 1, the PIV software components that are involved in and are subject to testing are as follows:

- PIV Middleware that implements the functions in the PIV Client API and interfaces with the PIV Card Application (resident on the card) by generating commands (APDUs) to the PIV card command interface.
- PIV Card Application that implements the PIV Card Application card command interface, accesses and modifies the content of PIV data objects, and facilitates realization of PIV authentication use cases.

3.1 PIV Middleware Tests

These tests will validate that the PIV Middleware conforms to the specification in Part 3 of SP 800-73-3. Conformance criteria includes correct implementation of all functions in the PIV client API, generation of appropriate commands for the PIV card command interface to communicate with the PIV Card Application and return the prescribed response codes to the calling agency application. This test, however, does not validate the functional requirements or the testing of the FIPS 201-mandated card application parameters, which are covered under the PIV data model tests.

The following client API functions are tested for conformance:

- pivConnect
- pivDisconnect
- pivSelectCardApplication
- pivLogIntoCardApplication
- pivGetData
- pivLogoutOfCardApplication
- pivCrypt
- pivPutData
- pivGenerateKeyPair
- pivMiddlewareVersion

These functions will be tested for their response to both the valid and the error conditions as defined by this document. To conduct these tests, a smart card with an "SP 800-73-3-conformant PIV Card Application" (refer to section 7.2 for definition) must be accessible.

3.2 PIV Card Application Tests

PIV Card Application tests cover the following:

- The PIV Card Application card command interface as per Part 2 of SP 800-73-3, including the security conditions for executing each command in the interface as well as the security conditions for accessing and storing each of the associated data objects.
- Presence of all mandatory data objects as well as accessibility and storage of all implemented data objects using the identifiers specified in Part 1 of SP 800-73-3.

The tests are performed through test scripts communicating directly with a PIV Card through the API of the driver that comes with the card reader.

3.2.1 PIV Card Application Card Command Interface Tests

These tests will validate that the card under test can successfully execute the commands in the PIV card command interface. Successful execution constitutes the card responding with appropriate data and response status codes to the commands sent by the test system. It also involves setting state variables within the PIV Card. For example, the criteria for successful execution of the SELECT command involve the following:

- The Response Status Code returned is "90 00".
- The application property template is returned with the right format and content.
- The "PIV Card Application" is the value of "currently selected application" (state variable) on the card.

The card command interface test suite includes conformance tests for the following PIV Card Application commands:

- Data access commands
  - SELECT
  - GET DATA
- Card authentication commands
  - VERIFY
  - CHANGE REFERENCE DATA
  - RESET RETRY COUNTER
  - GENERAL AUTHENTICATE
- Credential initialization and administration commands
  - PUT DATA
  - GENERATE ASYMMETRIC KEY PAIR

The card edge commands will be validated against the following conditions:

- Card interface type (contact vs. contactless)
- Precondition for use (PIN verified, cryptographic authentication)
- Expected response status codes
- Appropriate state variables set in the card.

3.2.2 PIV Data Objects Accessibility and Storage Tests

The testing covers the following data objects

- The five mandatory data objects as defined in Part 1 of SP 800-73-3
  - Card Capability Container
  - Card Holder Unique Identifier (CHUID)
  - X.509 Certificate for PIV authentication
  - Cardholder Fingerprints
  - Security Object
- The twenty-eight optional data objects, also defined in Part 1 of SP 800-73-3
  - On-card Printed Information
  - Cardholder Facial Image
  - PIV Discovery Object
  - X.509 Certificate for Digital Signature
  - X.509 Certificate for Key Management
  - X.509 Certificate for Card Authentication
  - Key History Object
  - 20 retired X.509 Certificates for Key Management
  - Cardholder Iris Image

The data objects will be validated for the following conditions.

- Presence of all mandatory data objects and those optional objects in the vendor documentation.
- Accessibility and storage of data objects using the appropriate BER-TLV tags (specified identifiers –Chapter 4, Part 1 of SP 800-73-3).
- Appropriate container size allocations for each of the data objects.
- Data objects access rule (PIN vs. no PIN).
- Security condition for data objects storage (cryptographic authentication).

- Appropriate card interface type for accessing each of the data objects (contact vs. contactless).

4. Derived Test Requirements

DTRs show the type of tests required based on the specifications in SP 800-73-3. These specifications cover expected command behavior (in the case of interface specification), data object representation (in the case of PIV data model) and data contents (in the case of PIV authentication use cases).

Each DTR consists of the following:

- Actual condition statements taken/derived from the SP 800-73-3 specification – these include conditions for successful command execution for each command as well as exception behaviors explicitly called out through 'shall' statements in SP 800-73-3. Those exception behaviors that are implicit in SP 800-73-3 through listing of error codes associated with each command are tested only through Test Assertions (Appendices B, C and D) and are not part of the DTR condition statements. The condition statements are identified by codes starting with 'AS' followed by a running sequence that denotes the section in this document where they occur.
- Required Vendor Information – these include information that the vendors are mandated to provide in their documentation. The Required Vendor Information is identified by codes starting with 'VE' followed by a running sequence that denotes the section in this document where they occur.
- 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. It must be mentioned; however, that some of the required test procedures will not call out 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.

Validations of some DTRs are not covered by the test assertions provided in this document. These DTRs require compliance of the component with an external specification or standard such as ISO/IEC 7816 or ISO/IEC 14443. No required test procedures are provided for these DTRs, and a note is added to indicate that the assertion is externally tested. The tester checks the vendor documentation for claimed compliance with such requirement or the presence of an external test/compliance certificate obtained from the related standards testing body, when applicable.

Some DTRs cannot be validated through the test tools provided in this document. For example, the test tool cannot access the asymmetric private keys generated and stored on the card. Therefore, a note is added to indicate the assertion is not separately tested for these DTRs. The same note is added for DTRs that make general statements on the nature of the PIV Card and are validated as a result of the validation of many other DTRs. For example, the statement "each command that appears on the card command interface shall be implemented by a Card Application that is resident in the Integrated Circuit Card (ICC)" is validated through the entire card command interface test and does not require an individual test assertion.

5. 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 the purpose of the test, starting conditions and prerequisites, success criteria, and post-test conditions, when applicable. A list of test assertions can be seen in Appendices B and C.

The following three sets of test assertions are included in this document:

- PIV client API test assertions (see Section 3.1 for overview)
- PIV card command interface test assertions (per Section 3.2.1)
- PIV data objects accessibility and storage test assertions (per Section 3.2.2)

An overview of each of the above classes of test assertions is given in Sections 5.2 through 5.4.

5.1 Mapping from Test Categories to Test Assertions

All the DTRs in Appendix A conceptually come under one of the two broad categories of tests stated in Section 3 i.e., PIV Middleware tests and PIV Card Application tests. Similarly, each test assertion makes specific references to the related sections in SP 800-73-3 or the related DTRs. However, overall there is a many-to-many mapping from the test suite elements (individual tests) under each of these two broad categories of tests to the DTRs (i.e., one test can map to many DTRs and one DTR can map to many tests). A similar type of relationship exists between DTRs and test assertions. To narrow the search space for cross references, Table 5-1 presents a cross-referencing guide showing the relevant DTR sections (with the section in SP 800-73-3 document from which they were derived) and test assertion sections with respect to test classes in the two broad categories of tests.

Category/Classes of Test	DTR Section(s)	Test Assertion Section(s)
(1) PIV Middleware Tests (Section 3.1)	A.4 PIV End-Point Client API (Part 3 of SP 800-73-3)	Appendix B—PIV Client API Test Assertions
(2a) PIV Card Application Tests—PIV Card Application Card Command Interface Tests (Section 3.2.1)	(1) A.1 End-Point Concepts and Constructs (Ch 2, Part 2, of SP 800-73-3) (2) A.5 End-Point PIV Card Application Card Command Interface (Ch 3, Part 2 of SP 800-73-3)	Appendix C—PIV Card Command Interface Test Assertions
(2b) PIV Card Application Tests—PIV Data Object s Accessibility and Storage Tests (Section 3.2.2)	(1) A.2 End-Point PIV Data Objects Representation (Ch 4, Part 1 of SP 800-73-3) (2) A.3 End-Point Data Types and Their Representation (Ch 5, Part 1 of SP 800-73-3, and Part 2 of SP 800-73-3)	Appendix C—PIV Data Objects Accessibility and Storage Test Assertions

Table 5-1. Cross-referencing Guide

5.2 PIV Client API Test Assertions

This section provides conformance tests in the form of test assertions for the functions specified in Chapter 3, Part 3 of SP 800-73-3 (called client API) that the PIV Middleware is expected to support. The test assertions are described through a test assertions template. The template provides placeholders for describing the purpose of the test, the preconditions required to exercise the test, the parameter values used in test invocation, and the expected results as well as the state of the PIV system (value of state variables), if any, that will be affected by the test run (post-condition).

The conformance tests are run against the PIV Middleware, which in turn interacts with the PIV Card Application resident on the PIV Card. Hence, there are two pieces of software (PIV Middleware and PIV Card Application) that determine the outcome of each test run. Because the focus of the tests is the behavior of the PIV Middleware, the test configuration assumes the presence of a validated PIV Card Application.

The test assertions derived from SP 800-73-3, SP 800-78-2 are demonstrated by test cases in Appendix B and are summarized in Table 5-2.

Client API Command	Number of Test Cases (Appendix B)
pivConnect	6
pivDisconnect	4
picSelectCardApplication	4
pivLogIntoCardApplication	4
pivLogOutOfCardApplication	3
pivGetData[3]	40
pivPutData	58
pivGenerateKeyPair	8
pivCrypt	13
pivMiddlewareVersion	1

Table 5-2. List of Commands in Client API and Number of Test Cases

The PIV client API test cases are based on the following assumptions:

- There is a PIV Card with a validated PIV Card Application.
- A valid connection description is provided for the Card Application.
- A valid physical connection exists between an instance of the PIV card reader and the host where the PIV Middleware (client application) resides.
- No other application is currently connected to the PIV Card Application.

[3] A SP 800-73-3 conformant PIV Middleware has to be able to recognize and handle all PIV data object. Therefore, the number of test cases for pivGetData and pivPutData includes reading/writing all mandatory and optional PIV data objects.to/from a PIV card.

5.3 PIV Card Command Interface Test Assertions

This section provides conformance tests in the form of test assertions for the command set that is specified in Chapter 3, Part 2 of SP 800-73-3 (Card Application Card Command Interface) that the PIV Card Application is required to support. The test assertions are described through a test assertions template. The template provides placeholders for describing the purpose of the test, the preconditions required to exercise the test, the parameter values used in test invocation, and the expected results as well as the state of the PIV system (value of state variables), if any, that will be affected by the test run (post- condition).

The conformance tests are run to validate the PIV Card Application. Interaction with the PIV Card Application takes place through the API of the driver that comes with the card reader.

The list of commands in the PIV Card Application card command interface and the number of test cases, derived from test assertions in Appendix C are summarized in Table 5-3.

Card Application Command	Number of Test Cases (Appendix C)
SELECT	5 (Contact) + 3 (Contactless) = 8 (Total)
GET DATA[4]	38 (Contact) + 34 (Contactless) = 72 (Total)
VERIFY	7 (Contact) + 2 (Contactless) = 9 (Total)
CHANGE REFERENCE DATA	7 (Contact) + 2 (Contactless) = 9 (Total)
RESET RETRY COUNTER	6 (Contact) + 1 (Contactless) = 7 (Total)
GENERAL AUTHENTICATE	59 (Contact) + 59 (Contactless) = 118 (Total)
PUT DATA	66 (Contact) + 1 (Contactless) = 67 (Total)
GENERATE ASYMMETRIC KEY PAIR	7 (Contact) + 1 (Contactless) = 8 (Total)

Table 5-3. List of Commands in Card Application Command and Number of Test Cases

The following assumptions have been made with regard to the PIV Card command interface test cases:

- A valid PIV Card is inserted into the contact reader or placed near a contactless reader.
- A valid PC/SC connection exists between the test system and an instance of the reader.
- No application is currently connected to the PIV Card Application.
- No other contactless card is within the proximity of the contactless reader.

[4] For PUT DATA, GET DATA, GENERAL AUTHENTICATE and GENERATE ASYMMETRIC KEY PAIR positive and negative tests, the number of test cases is based on the assumption that mandatory and optional PIV data objects and asymmetric keys are implemented. The number of tests may be less, if he card application under test does not implement all optional data objects and asymmetric keys.

5.4 PIV Data Objects Accessibility and Storage Test Assertions

The following assumptions have been made with respect to the PIV data object representation test assertions:

- A PIV Card Application with a valid Application Identifier (AID) is resident on the card.
- The PIV Card Application is expected to have implemented all five mandatory PIV data objects of the PIV data model on the card.
- The presence of any one or more of the twenty-eight optional PIV data objects on the PIV Card is known from the vendor documentation.

6. Test and Compliance Documentation

There are two sets of compliance documentation: vendor required and test facility generated.

The vendor-required documents consist of the following:

- **Installation and Execution instructions (for PIV Middleware):** The vendor provides technical instructions and other documentation to aid the testing personnel in installing and using the PIV Middleware implementation under test. The PIV Middleware implementation could be in any higher-level programming language. Since all the implementations have to be tested from a common test program, the PIV Middleware vendor submitting the product for testing may have to provide wrapper programs in some cases to the test facility. The purpose of the wrapper program is to translate the test execution calls made using the test program to the PIV Middleware implementation's native program calls.

- **Technical documentation (for both PIV Card Application and PIV Middleware):** The vendor-supplied technical documentation must include the detailed technical description and the design of the implementation to be tested. This document includes, at a minimum, all the required vendor information specified in DTRs in Appendix A of this document.

- **Security-related information:** Security-related information such as the PIV Card Application PIN, the PIN unblocking PIN, the number of unsuccessful attempts the PIV Card Application will allow using the wrong Card Application and PIN unblocking PINs, the optional cryptographic algorithms supported by the PIV Card Application, etc. should be provided by the vendor.

The test facility-generated documents are required for performing and reporting the test process. The following are some of the examples:

- **Checklists.** Checklists provide the tester with a list of actions and requirements to complete before the test starts. Information required in the preconditions section of the assertions is included in the checklists.

- **Test logs.** A test log is kept for each test run on any component and is used to summarize the results of all the tests run.

- **Test reports.** These provide the background (environmental information) for each of the test cases as well as summary of outcomes from test runs (from test logs) associated with each test case.

A test case is a sequence of command/function invocations that pertain to a given execution condition for the 'command/function under test'. For example, if the GET DATA command is the command/function under test, then the execution condition 'Invocation of this function after PIN verification' will consist of the following sequence of command/function invocations – SELECT, VERIFY, GET DATA, and collectively constitutes a test case. There may be many test runs for this test case. The function invocations returning the expected return codes for a test case in all test runs indicates that the command/function has been implemented correctly.

7. Acceptance Criteria

Acceptance criteria are based on the compliance of the item under test with the requirements defined in FIPS 201 and the accompanying special publication documents. The criteria are further specified in the following sections, based on the type of test being conducted.

7.1 Acceptance Criteria for the PIV Middleware Test

The PIV Middleware test acceptance criteria will be based on the middleware application under test passing all the PIV client API test assertions. The middleware should return appropriate return codes in response to executing the client API functions as defined in Section 3, Part 3of SP 800-73-3. The middleware should also be able to send the correct card commands to and interpret the responses received from the "SP 800-73-3 conformant PIV Card Application" (refer to section 7.2 for definition). The test assertions detail the pass/fail criteria defined for each test case that is designed to test a certain condition being tested.

7.2 Acceptance Criteria for the PIV Card Application Tests

Acceptance criteria for the PIV Card Application tests are based on the PIV Card Application passing the following two classes of tests: PIV Card Application card command interface tests and PIV data objects accessibility and storage tests. The PIV Card application that has passed these classes of tests is called "SP 800-73-3 conformant PIV Card application."

For PIV Card Application card command tests, the PIV Card Application should send the appropriate response status codes and application data in response to commands. It should also set or reset certain card state variables and thus fulfill the test post conditions.

For the PIV data objects accessibility and storage tests, the PIV Card Application should show the presence of all mandatory PIV data objects and published optional PIV data objects. It should also demonstrate the ability to access and store all the above data objects using the correct BER-TLV tag under the appropriate security conditions and interfaces (contract or contactless) and that the containers for storing them have the needed size requirements.

The acceptance criteria for the testing of PIV Card functionalities, for which FIPS 201 makes reference to external documents (such as digital signature formats), is based on visual verification of vendor-provided documents and test/compliance certificates.

8. Test and Compliance Process

The PIV software component that passes all applicable tests, as explained in this document, will be considered conformant. This document provides the technical details for the testing of the two PIV software components. In this context, compliance means—

- Passing the related test assertions explained in this document
- Passing the inspection/verification of the required vendor documentation.

The certified and/or accredited test laboratory that will conduct the testing has the following responsibilities—

- Prepare and provide the test application forms and the documentation
- Receive and configure the PIV software component to be tested
- Conduct the test with a testing toolkit
- Review the test results and report failures
- Inspect the vendor documentation
- Communicate the results.

Upon vendor's submission of the request for PIV component certification, the required documentation, and the PIV software components to be tested, the test laboratory configures the test system, records all preconditions, and runs the applicable suite of tests for the submitted PIV component. After conducting the tests, the test laboratory evaluates the test results and communicates the Test Results Summary (TRS) and Test Run Details (TRD) to the vendor.

The Test Results Summary provides the overall environmental information (date and time the tests were conducted, the tester name and vendor product identifier, etc.) as well as the summary conclusion for tests associated with that particular class. The format of the summary report will vary depending upon the test classes. The sample format for each of the above types of summary reports is given in Appendix F. The TRS associated with each of the three classes are:

+ PIV Client API Test Summary
+ Card Command Interface Test Summary
+ PIV Data Objects Accessibility and Storage Test Summary

The TRD are used to log the details of each test run associated with each of the three classes in the test suite. They provide the details of the outcome of each test run for various execution conditions. This detailed report will enable the product vendor to make the necessary logic changes to the implementation of the various commands/interfaces and data object representations in order to become fully conformant.

8.1 Failure Review

The test will be repeated once for components that do not pass the tests. After the retest, the tester prepares for each failure a discrepancy report that summarizes the purpose of the test, the progression of steps, and the responses received from the tested components. The discrepancy report will be internally reviewed and discussed by the test lab before an official response is sent to the vendor. Vendors who object to the results presented in the discrepancy report must explain their reason for the objection. If the reason necessitates another retest, the test laboratory may consider repeating the test. Otherwise, the test lab will seek the guidance of the NIST personnel on the failure, before the component is returned to the vendor to be corrected.

Appendix A—Derived Test Requirements

A.1 End-Point Concepts and Constructs (Chapter 2, Part 2 of SP 800-73-3)

A.1.1 Platform Requirements

AS01.01: The PIV Card Application shall place the following requirements on the ICC platform on which it is implemented or installed:

- **Global security status that includes the security status of a global cardholder PIN**
- **Application selection using a truncated Application Identifier (AID)**
- **Ability to reset the security status of an individual application**
- **Indication to applications as to which physical communication interface – contact versus contactless- is in use**
- **Support for the default selection of an application upon warm or cold reset.**

Note: This assertion is not separately tested.

A.1.2 Card Applications

AS01.02: Each command that appears on the card command interface shall be implemented by a card application that is resident in the ICC.

Note: This assertion is not separately tested – collection of DTRs for all commands implicitly tests this assertion.

AS01.03: Each card application shall have a globally unique name called its AID [ISO/IEC 7816, Part 4].

Note: This assertion is tested as part of the AS05.05 through AS05.10.

AS01.04: Except for the default applications, **access to the card commands and data objects of a card application shall be gained by selecting the card application using its application identifier.**

Note: This assertion is tested as part of AS05.11.

AS01.05: The Proprietary Identifier eXtension (PIX) of the AID shall contain an encoding of the version of the card application.

Note: This assertion is tested as part of the AS05.05 through AS05.10.

A.1.2.1 Personal Identity Verification Card Application

AS01.06: The AID of the Personal Identity Verification Card Application (PIV Card Application) shall be: 'A0 00 00 03 08 00 00 10 00 01 00'

Note: This assertion is tested as part of the AS05.05 through AS05.10.

AS01.07: For the first version of the PIV Card Application, the AID shall consist of the NIST Registered application provider IDentifier (RID) 'A0 00 00 03 08' followed by the application portion of the NIST PIX indicating the PIV Card Application '00 00 10 00' and then the version portion of the NIST PIX '01 00'.

Note: This assertion is tested as part of the AS05.05 through AS05.10.

A.1.2.2 Default Selected Card Application

AS01.08: The card platform shall support a default selected card application. In other words, there shall be a currently selected application immediately after a cold or warm rest.

Required Vendor Information

VE01.08.01: The vendor shall specify in its documentation the default selected card application.

Required Test Procedures

TE01.08.01: The tester shall validate that there is a default selected card application which is the one specified by the vendor in VE01.08.01.

A.1.3 Security Architecture

A.1.3.1 Access Control Rule

AS01.09: The access control rule shall consist of an access mode and a security condition.

Note: This assertion is not separately tested.

AS01.10: The action described by the access mode can be performed on the data object if and only if the security condition evaluates to TRUE for the current values of the security status.

Note: This assertion is not separately tested.

AS01.11: If there is no access control rule with an access mode describing a particular action, then that action shall never be performed on the data object.

Note: This assertion is not separately tested.

A.1.3.2 Security Status

AS01.12: Associated with each authenticatable entity shall be a set of one or more Boolean variables each called a security status indicator of the authenticatable entity.

Note: The security status indicators will be tested indirectly through the functional testing.

AS01.13: The security status indicator of an authenticatable entity shall be TRUE if the credentials associated with the security status indicator of the authenticatable entity have been authenticated and FALSE otherwise.

Note: The security status indicators will be tested indirectly through the functional testing.

AS01.14: The successful execution of an authentication protocol shall set the security status indicator associated with the credentials used in the protocol to TRUE.

Note: The security status indicators will be tested indirectly through the functional testing.

AS01.15: A security status indicator shall be said to be a global security status indicator if it not changed when the currently selected application changes from one application to another.

Note: This assertion is not separately tested.

AS01.16: A security status indicator is said to be an application security status indicator if it is set to FALSE when the currently selected application changes from one application to another.

Required Vendor Information

VE01.16.01: The vendor shall specify in its documentation that the application security status indicators will be set to FALSE when the currently selected application changes from one application to another.

Required Test Procedures

TE01.16.01: The tester shall visually validate that the vendor documentation contains the requirement stated in VE01.16.01.

A.1.3.3 Authentication of an Individual

AS01.17: Personal identification numbers (PIV Card Application PINs and PUKs) **presented to the card command interface shall be 8 bytes long.**

Note: This assertion is tested as part of AS05.22A.

AS01.18: If the actual PIN length is less than 8 bytes, it shall be padded to 8 bytes with 'FF' and appended to the actual PIN. The bytes comprising the PIV Card Application PIN shall be limited to values 0x30 – 0x39, the ASCII values for the decimal digits '0' – '9'. The bytes comprising the PUK shall be limited to the values 0x00 – 0xFE (i.e., shall not include 'FF'). If the Global PIN is used by the PIV Card Application then the above encoding, length, and padding requirements for the PIV Card Application PIN shall apply to the Global PIN.

Note: This assertion is tested as part of AS05.22A.

A.1.4 PIV Card Application Status Variables

AS01.19: When the PIV Card Application is the currently selected application, the following status variables shall be associated with it.

- **Status Variable: Global Security Status Indicators– must always be defined. Can be used by all applications on the card platform. Maintained by: card platform.**
- **Status Variable: Currently selected application – must always be defined. The platform shall support the selection of a card application using a possibly right-truncated application identifier and there shall always be a currently selected application. Maintained by: card platform.**
- **Status Variable: Application security status Indicators – must always be defined. These indicators are local to the PIV Card Application. Maintained by: PIV Card Application.**

Note: This assertion is not separately tested.

A 1.5 Card Platform Configuration

AS01.20: Both single-chip/dual-interface and dual-chip implementations shall be feasible.

Note: This assertion is not separately tested.

AS01.21: In the single-chip/dual-interface configuration, the PIV Card Application shall be provided the information regarding which interface is in use.

Required Vendor Information

VE01.21.01: The card operating system should inform the PIV Card Application the communication interface in use.

Required Test Procedures

TE01.21.01: The tester shall validate that the card platform informs the PIV Card Application of the interface being used.

Note: This assertion is not separately tested. This assertion is indirectly tested by verifying whether the card application returns '6A 81' for those commands that cannot be exercised through contactless interface. The tester shall verify response code '6A 81' is returned.

TE01.21.02: The tester shall validate that the PIV Card Application checks that a contact interface is being used for contact-only APDUs.

Note: This assertion is not separately tested. This assertion is indirectly tested by verifying whether the card application returns '6A 81' for those commands that cannot be exercised through contactless interface. The tester shall verify that response code '6A 81' is returned.

AS01.22: In the dual-chip configuration, a separate PIV Card Application shall be loaded on each chip.

Note: This assertion is not separately tested.

A.2 End-Point Data Objects (Part 1, Chapter 4 of SP 800-73-3)

A.2.1 PIV Card Application Data Objects

AS02.01: A PIV Card Application shall contain five mandatory data objects and twenty-eight optional data object for interoperable use.

- **The five mandatory data objects are the following: 1. Card Capability Container 2. Card Holder Unique Identifier 3. X.509 Certificate for PIV Authentication 4. Cardholder Fingerprints 5. Security Object**
- **The twenty-eight optional data objects for interoperable use are the following: 1. Cardholder Facial Image 2. Printed Information 3. X.509 Certificate for Digital Signature 4. X.509 Certificate for Key Management 5. X.509 Certificate for Card Authentication 6. Discovery Object 7. Key History Object 8. twenty retired X.509 Certificates for Key Management and 9. Cardholder Iris Image**

Note: This assertion is not separately tested.

A.2.2 OIDs and Tags of PIV Card Application Data Objects

AS02.02: For the purpose of constructing PIV Card Application data object names in the CardApplicationURL in Card Capability Container (CCC) of the PIV Card Application, the NIST RID ('A0 00 00 03 08') shall be used and the card application type shall be set to '00'.

Required Test Procedures

Note: This assertion is tested as part of AS02.03.

AS02.03: For all data objects present on the card, the object identifiers (OIDs) used by PIV Client Application to refer to them, and associated BER-TLV tags used by PIV Card Command Interface shall conform to the entries in Table 2, Part 1 of SP 800-73-2.

Required Vendor Information

VE02.03.01: The vendor shall state in its documentation the list of all the data objects present on the card along with the OIDs and BER-TLV tags associated with them.

Required Test Procedures

TE02.03.01: The tester shall validate that the OIDs and BER-TLV tags of all the data objects present on the card conform to the Table 2, Part 1 of SP 800-73-3, and accurately represent the actual data objects observed by the tester as being implemented on the card.

A.3 End-Point Data Types and Their Representations

A.3.1 Algorithm Identifier

AS03.01: The algorithm identifiers for the cryptographic algorithms implemented on the card shall conform to entries in Table 6-2 of SP 800-78-2.

Required Vendor Information

VE03.01.01: The vendor shall state the identifiers associated with all the algorithms supported by the card.

Required Test Procedures

TE03.01.01: The tester shall validate the presence of all algorithm identifiers implemented on the vendor documentation and the card, and that they comply with Tables 3-1, 5.1 and 6.2 of SP 800-78-2.

A.3.2 Application Property Template

AS03.02: Upon selection, the PIV Card Application shall return the application property template described in tables 3 and 4, Part 2 of SP 800-73-3.

Required Vendor Information

VE03.02.01: The vendor shall provide in its documentation the PIV card application property template along with their TLVs.

Required Test Procedures

TE03.02.01: The tester shall visually validate that the information provided in response to VE03.02.01 is in conformance with tables 3 and 4, Part 2 of SP 800-73-3.

TE03.02.02: The tester shall validate that the information provided in VE03.02.01 is actually implemented by the card.

A.3.3 Authenticator

AS03.03: The authenticator BER-TLV used on the PIV client application programming interface shall have the structure described in Table 3, Part 3 of SP 800-73-3.

Required Vendor Information

VE03.03.01: The vendor shall provide a list of all the authenticators along with their tags and possible values, when applicable.

Required Test Procedures

TE03.03.01: The tester shall visually validate that the vendor documentation states the correct tags for the "reference data" and "Key Reference" as shown in Table 3, Part 3 of SP 800-73-3.

TE03.03.02: The tester shall validate that the card returns the correct tags and values in the authenticator data object as specified in Table 3, Part 3 of SP 800-73-3.

A.3.4 Connection Description

AS03.04: The connection description BER-TLV used on the PIV client application programming interface shall have the structure described in table 2, Part 3 of SP 800-73-3.

Required Vendor Information

VE03.04.01: The vendor shall provide in its documentation the format and content of the

Connection Description Templates implemented by the card.

Required Test Procedures

TE03.04.01: The tester shall validate the presence of the information provided in VE03.04.01 and that the Connection Description Template sent to the card conforms to Table 2, Part 3 of SP 800-73-3.

AS03.05: At most one selection from the '8x' series and one selection from the '9x' series shall appear in the connection description template as specified in AS03.04.

Note: This assertion is tested as part of AS03.04.

A.3.5 Key References

AS03.06: The key reference, when represented as a byte, occupies b8 and b5-b1 while b7 and b6 shall be set to 0.

Note: This assertion is not separately tested.

AS03.07: The key references used on all PIV interfaces shall be from the list found in Table 6-1 of SP 800-78-2 (for cryptographic key references) or Table 3, Part 1 of SP 800-73-3 (for PIN references).

Note: This assertion is not separately tested.

AS03.08: The only key reference allowed in the P1 parameter of the RESET RETRY COUNTER command is the PIV Card Application PIN.

Required Vendor Information

VE03.08.01: The vendor shall specify in its documentation that the card conforms to the assertion stated in AS03.08.

Required Test Procedures

TE03.08.01: The tester shall select the PIV Card Application and attempt to reset the Global PIN retry counter with the RESET RETRY Counter command and validate that the global PIN retry counter was not reset.

TE03.08.02: The tester shall visually validate the presence of the information required in VE03.08.01.

Note: This assertion is not separately tested.

A.3.6 Status Words

AS03.09: A status word shall be a 2-byte value returned by an entry point on the client application programming interface or a card command at the card edge.

Note: This assertion is not separately tested – since it is part of the process of testing status words (return codes) for every command in PIV Client API and PIV card command interfaces.

AS03.010: Recognized values of all SW1-SW2 pairs used as return values on both the PIV client application programming and PIV card command interfaces shall be from the list provided in Table 5, Part 1 of SP 800-73-3.

Note: This assertion is not separately tested.

AS03.011: A data object shall be identified on the PIV client application programming interface using its OID.

Note: This assertion is not separately tested.

AS03.012: An object identifier on the PIV client application programming interface shall be a dot-delimited string of the integer components of the OID.

Note: This assertion is not separately tested.

AS03.013: A data object shall be identified on the PIV Card Application card command interface using its BER-TLV tag.

Note: This assertion is not separately tested.

A.4 End-Point Client Application Programming Interface (Part 3 of SP 800-73-3)

AS04.01: Entry points on the PIV client application programming interface shall include all functions listed in Table 1, Part 3 of SP 800-73-3.

Note: This assertion is tested as part of AS04.02 through AS04.11.

A.4.1 Entry Points for Communication

Required Vendor Information & Required Test Procedures

To test the entry points or commands that should be supported by client application, the only information that the vendor has to provide is the PIV Card Application version that the client application supports. All parameter values for exercising the commands have to be obtained from the PIV Card Application vendor documentation, using the mapping of client application

entry point commands to the PIV Card Application card commands. This mapping is given in Table A-1 below. Hence this section contains only tester requirements in terms of Required Test Procedures.

Client Application Entry Points	PIV Card Application Card Command[5]	Mapping Description
pivConnect	No equivalent command	For establishing a connection session with card reader.
pivDisconnect	No equivalent command	For disconnecting a connection session with the card reader
pivSelectCardApplication	SELECT	Passes the AID value. Sets the value for 'Currently Selected Application' on the PIV card. Establishes the PIV Card Application security status.
pivLogIntoCardApplication	VERIFY	Provides the key reference for PIV Card Application PIN as well as the PIN string and passes. Sets/Updates the PIV Card Application Security Status on the card
pivLogOutOfCardApplication	RESET (not specified in SP 800-73-3)	Resets the PIV Card Application Security Status on the card
pivGetData	GET DATA	Maps the Object Identifier (OID) to BER-TLV tag for the selected object
pivPutData	PUT DATA	Maps the OID to BER-TLV tag for the selected object
pivGenerateKeyPair	GENERATE ASYMMETRIC KEY PAIR	Passes the Key Reference and Cryptographic Mechanism identifier values
pivCrypt	GENERAL AUTHENTICATE	Passes the Key Reference, Cryptographic Algorithm reference and the string to be acted upon. Sets/Updates the PIV Card Application Security Status on the card
pivMiddlewareVersion	No equivalent command	Returns the PIV Middleware Version supported by the PIV Middleware IUT

Table A-1 PIV Command Mapping

A.4.1.1 pivConnect

AS04.02: The pivConnect's purpose is to connect the client application programming interface and hence the client application itself to the PIV Card Application on a

[5] It is assumed that some of these functions will use GET RESPONSE and chaining to accomplish the read or write to the card.

specific ICC.

TE04.02.01: The tester shall validate that the client application implements the pivConnect as per SP 800-73-3, Part 3.

A.4.1.2 pivDisconnect

AS04.03: The pivDisconnect's purpose is to disconnect the client application programming interface from the PIV Card Application and the ICC containing the PIV Card Application.

TE04.03.01: The tester shall validate that the client application implements the pivDisconnect as per SP 800-73-3 Part 3.

A.4.2 Entry Points for Data Access

A.4.2.1 pivSelectCardApplication

AS04.04: The pivSelectCardApplication sets the currently selected card application and establishes the PIV Card Application security state.

TE04.04.01: The tester shall validate that the client application implements the pivSelectCardApplication as per SP 800-73-3, Part 3.

A.4.2.2 pivLogIntoCardApplication

AS04.05: The pivLogIntoCardApplication sets/updates the security state within the PIV Card Application.

TE04.05.01: The tester shall validate that the client application implements the pivLogIntoCardApplication as per SP 800-73-3 Part 3.

A.4.2.3 pivGetData

AS04.06: The pivGetData returns the entire data content of the named data object.

TE04.06.01: The tester shall validate that the client application implements the pivGetData as per SP 800-73-3 Part 3.

A.4.2.4 pivLogoutOfCardApplication

AS04.07: The pivLogoutOfCardApplication resets the application security state of the PIV Card Application.

TE04.07.01: The tester shall validate that the client application implements the pivLogoutOfCardApplication as per SP 800-73-3 Part 3.

A.4.3 Entry Points for Cryptographic Operations

A.4.3.1 pivCrypt

AS04.08: pivCrypt perform a cryptographic operation such as encryption or signing on a sequence of bytes.

TE04.08.01: The tester shall validate that the client application implements the pivCrypt as per SP 800-73-3 Part 3.

A.4.4 Entry Points for Credential Initialization and Administration

A.4.4.1 pivPutData

AS04.09: The pivPutData replaces the entire data content of the named data object with the provided data.

TE04.09.01: The tester shall validate that the client application implements the pivPutData as per SP 800-73-3 Part 3.

A.4.4.2 pivGenerateKeyPair

AS04.10: The pivGenerateKeyPair generates an asymmetric key pair in the currently selected application.

TE04.10.01: The tester shall validate that the client application implements the pivGenerateKeyPair as per SP 800-73-3 Part 3.

A.4.4.3 pivMiddlewareVersion

AS04.11: SP 800-73-3 Part 3 conformant PIV Middleware shall implement all PIV Middleware functions listed in Table 1 of SP 800-73-3 and be able to recognize and process all mandatory and optional PIV data objects. The pivMiddlewareVersion returns the PIV Middleware version string. For SP 800-73-3 Part 3 conformant PIV Middleware, the parameter returns "800-73-3 Client API".

TE04.11.01: The tester shall validate that the client application supports all functions listed in table 1 of SP 800-73-3 and implements the pivMiddlewareVersion as per SP 800-73-3 Part 3 by returning the "800-73-3 Client API" parameter string.

A.5 End-Point PIV Card Application Card Command Interface (Part 2 of SP 800-73-3)

AS05.01: All PIV Card Application card commands listed in Table 2, Part 2 of SP 800-73-3 shall be supported by a PIV Card Application.

Required Vendor Information

VE05.01.01: The vendor shall provide the list of all PIV Card Application card commands, along with the interface(s) (contact or contactless) they support, the security condition(s) they are subject to and their support for command chaining as implemented by the card.

Required Test Procedures

TE05.01.01: The tester shall validate that the information presented in response to VE05.01.01 by the vendor complies with Table 2, Part 2 of SP 800-73-3.

TE05.01.02: The tester shall validate that the card implements all the commands as required in Table 2, Part 2 of SP 800-73-3.

TE05.01.03: The tester shall validate that the commands are implemented only through the interfaces allowed as shown in Table 2, Part 2 of SP 800-73-3.

TE05.01.04: The tester shall validate that the commands are implemented only after the security condition associated with them are satisfied, as shown in the table, via the specified interface.

TE05.01.05: The tester shall validate that only the commands as indicated in the table are allowed for chaining via the interface supported after the security condition is satisfied.

AS05.02: Card commands indicated with a 'Yes' in the Command Chaining column shall support command chaining for transmitting a data string too long for a single command as defined in ISO/IEC 7816-4 [6].

Note: This assertion is tested as part of AS05.01.

AS05.03: The PIV Card Application shall return the status word of '6A81' (Function not supported) when it receives a card command on the contactless interface marked "No" in the Contactless Interface column in the table in AS05.01.

Note: This assertion is tested as part of AS05.01.

AS05.04: Cryptographic protocols using private/secret keys requiring PIN security condition shall not be used on the contactless interface**.**

Note: This assertion is tested as part of AS05.01.

A.5.1 PIV Card Application Card Commands for Data Access

A.5.1.1 SELECT Card Command

AS05.05: The PIV Card Application shall be selected by providing its application identifier ‘A0 00 00 03 08 00 00 10 00 vv vv’ in the data field of the SELECT command where 'vv vv' is the version of the PIV Card Application to be made the currently selected application.

Required Vendor Information

VE05.05.01: The vendor shall specify in its documentation the PIV Card Application Identifier.

Required Test Procedures

TE05.05.01: The tester shall validate that the PIV Card Application is selected by providing its application identifier as specified in VE05.05.01.

AS05.06: There shall be at most one PIV Card Application on any ICC.

Required Vendor Information

VE05.06.01: The vendor shall state in its documentation that there is only one PIV Card Application on the ICC.

Required Test Procedures

TE05.06.01: The tester shall visually validate the vendor documentation for the information provided in VE05.06.01.

AS05.07: The PIV Card Application can also be made the currently selected application by providing a right-truncated version; that is, without the two-byte version number, 'vv vv'; in the data field of the SELECT command 'A0 00 00 03 08 00 00 10 00’

Required Vendor Information

VE05.07.01: The vendor shall specify in its documentation whether the card implements the application selection by the right-truncated version.

Required Test Procedures

TE05.07.01: The tester shall visually validate that the information in VE05.07.01 is present in the vendor documentation.

TE05.07.02: The tester shall validate that the PIV application is selectable by the right-truncated SELECT command.

AS05.08: The complete AID, including the two-byte version, of the PIV Card Application that became the currently selected application upon successful execution of the SELECT command shall be returned in the application property template.

Note: This assertion is tested as part of AS03.02.

AS05.09: If the currently selected application is the PIV Card Application when the SELECT APPLICATION command is given and the AID in the data field of the SELECT APPLICATION is either the AID of the PIV Card Application or its right-truncated version thereof, then the PIV Card Application shall continue to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall be unchanged.

Required Vendor Information

VE05.09.01: The vendor shall provide information in its documentation validating the compliance with the statement in AS05.09.

Required Test Procedures

TE05.09.01: The tester shall validate that when the currently selected application is the PIV Card Application and the SELECT command is sent with an AID that is either the AID of the PIV Card Application or its right-truncated version, then the PIV Card Application shall continue to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall be unchanged.

AS.05.10 If the currently selected application is the PIV Card Application when the SELECT APPLICATION command is given and the AID in the data field of the SELECT APPLICATION is an invalid AID not supported by the PIV card then the PIV Card Application shall continue to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall remain unchanged.

Required Vendor Information

VE05.10.01: The vendor shall provide information in its documentation validating the compliance with the statement in AS05.10.

Required Test Procedures

TE05.10.01: The tester shall validate that when the currently selected application is the PIV Card Application, and the SELECT APPLICATION command is sent with an AID that is not a valid AID supported by the card, then the PIV Card Application continues to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall be unchanged.

AS.05.11: If the currently selected application is the PIV Card Application when the SELECT command is given and the AID in the data field of the SELECT APPLICATION is not the PIV card Application (nor the right-truncated version thereof), but a valid AID supported by the ICC, then the PIV Card Application shall be deselected and all PIV Card Application security status indicators shall be set to FALSE.

Required Vendor Information

VE05.11.01: If the ICC supports another card application, the vendor shall provide information in its documentation validating the compliance with the statement in AS05.11.

Required Test Procedures

TE05.11.01: If the ICC supports another card application, the tester shall validate that when the currently selected application is the PIV Card Application and the SELECT APPLICATION command is sent with an AID different from the PIV Card Application AID (or its right-truncated version), but is a valid AID supported by the ICC, then PIV Card Application shall be deselected and its security status indicators shall be set to FALSE.

A.5.1.2 GET DATA Card Command

AS.05.12 The GET DATA card command retrieves the data content of the single data object whose tag is given in the data field.

Note: This assertion is tested as part of AS05.01.

AS05.12A: The GET DATA card command retrieves the data content of the data object only after the access rule associated with the data object (Appendix A, Table 6, Part 1 of SP 800-73-3) evaluates to TRUE.

Required Vendor Information

VE05.12A.01: The vendor shall specify in its documentation the access rule for each of the data objects or make a reference to Table 6 in Appendix A, Part 1 of SP 800-73-3.

Required Test Procedures

TE05.12A.01: For implementations without the Discovery Object or implementations with the Discovery Object implemented and the PIN usage policy's first byte set to 0x40: The Tester shall validate that all data objects that require a PIN shall only be accessible after a successful validation of the PIV Card Application PIN through the VERIFY command.

TE05.12A.02: For implementations with the Discovery Object implemented and the PIN usage policy first byte set to 0x60: 1) The Tester shall validate that all data objects that require a PIN

shall be accessible after a successful VERIFY with the PIV Card Application PIN. 2) The Tester also shall validate that all data objects that require a PIN shall be accessible after a successful VERIFY with the Global PIN.

TE05.12A.03: The Tester shall validate that all data objects whose access rule is "Always Read" shall be accessible with or without PIV Card Application PIN validation or Global PIN validation (if implemented as indicated in the Discovery Object).

A.5.2 PIV Card Application Card Commands for Authentication

A.5.2.1 VERIFY Card Command

AS.05.13: PIV Card Application that satisfy the PIV ACRs for PIV data object access and command execution with both PIV Card Application PIN and Global PIN, shall implement the discovery object with the PIN Usage Policy set to 0x60 zz where zz is set to either 0x10 or 0x20.

Required Vendor Information

VE05.13.01: The vendor shall confirm that the PIV Card Application PIN can be used for PIV data object access and command execution. If the Global PIN (in addition to the PIV Card Application PIN) is used for data access and command execution while the PIV Card Application is the currently selected application, the vendor shall state in its documentation that the card supports the assertion made in AS05.13.

Required Test Procedures

TE05.13.01: The tester shall validate that the PIV Card Application PIN can be used for PIV data object access and command execution. The tester shall validate that when the Global PIN satisfies the PIV ACRs for PIV data object access and command execution with both PIV Card Application PIN and Global PIN then: 1) the Discovery Object is implemented with the PIN Usage Policy set to 0x60 zz, where zz is set to either 0x10 or 0x20 and 2) the Global PIN can be used for PIV data object access and command execution.

AS.05.14: Key reference '80' specific to the PIV Card Application (i.e., local key references) and, optionally, the Global PIN with key reference '00' are the only key references that may be verified by the PIV Card Application's VERIFY command.

Note: This assertion is tested as part of AS05.013.

AS.05.15: Key reference '80' shall be verified by the PIV Card Application VERIFY command.

Note: This assertion is tested as part of AS05.13.

AS.05.16: If the PIV Card Application contains the Discovery Object as described in Part 1 of SP 800-73-3, and the first byte of the PIN Usage Policy value is set to 0x60, then key reference '00' shall be able to be verified by the PIV Card Application VERIFY command.

Required Vendor Information

VE05.16.01: The vendor shall specify in its documentation if the Global PIN is implemented with the VERIFY command to satisfy access control rules to read PIN protected PIV data objects. If implemented, the vendor shall also specify the Discovery Object to be present on card with the first byte of the PIN Usage Policy value set to 0x60.

Required Test Procedures

TE05.16.01: The tester shall validate that if global PIN is implemented with the VERIFY command, then the Discovery Object is present on the card with the first byte of the PIN Usage Policy value set to 0x60.

AS.05.17: If the current value of the retry counter associated with the key reference is zero, then the comparison shall not be made and the PIV Card Application shall return the status word '69 83' (Authentication method blocked).

Required Vendor Information

VE05.17.01: The vendor shall specify in its documentation the reset value of the retry counters associated with all the key references implemented on the card.

Required Test Procedures

TE05.17.01: The tester shall validate that the PIV Card Application returns '69 83' (Authentication method blocked) in response to the VERIFY command, when the retry counter associated with the key reference is zero.

AS05.18: If the authentication data in the command data field does not satisfy the criteria in Section 2.4.3, then the card command shall fail, and the PIV Card Application shall return the status word '6A 80'.

Required Vendor Information

VE05.18.01: The vendor shall specify in its documentation the conditions (and associated status word) when the command will fail.

Required Test Procedures

TE05.18.01: The tester shall validate that when the authentication data in the command data field does not satisfy the criteria in Section 2.4.3, SP 800-73-3 Part 1, the card command fails, and the PIV Card Application returns the status word '6A 80'.

AS05.19: If the authentication data in the command data field does not match reference data associated with the key reference then the card command shall fail.

Required Vendor Information

VE05.19.01: The vendor shall specify in its documentation the conditions (and associated status word) when the command will fail.

Required Test Procedures

TE05.19.01: The tester shall validate that when the authentication data in the command data field does not match the reference data, the card command fails, and the PIV Card Application returns the status word '6A 80.

AS05.20 If the card command succeeds, then the security status of the key reference shall be set to TRUE and the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference.

Required Vendor Information

VE05.20.01: Same as VE05.17.01.

Required Test Procedures

TE05.20.01: The tester shall validate that the retry counter associated with the key reference shall be set to the reset retry value specified by the vendor in VE05.20.01 (not decremented), when the VERIFY command succeeds.

AS05.21: If the card command fails, then the security status of the key reference shall be set to FALSE and the retry counter associated with the key reference shall be decremented by one.

Required Vendor Information

VE05.21.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.21.

Required Test Procedures

TE05.21.01: The tester shall validate that when the VERIFY command fails, the retry counter associated with the key reference is decremented by one.

AS05.22A: If the PIN value in the reference data field of the command field is not padded to 8 bytes, the PIV Card Application shall return the status word '6A 80'.

Required Vendor Information

VE05.22.01A: The vendor shall state in its documentation that the card supports the assertion made in AS05.22A.

Required Test Procedures

TE05.22.01A: The tester shall validate that the vendor documentation contains the information required in VE05.22.01A and the card returns status word ‘6A 80’, when the PIN information in the reference data field of the command is not padded to 8 bytes.

AS05.22B: If the key reference is set to a value other than what is supported by the card, the PIV Card Application shall return the status word ‘6A 88’ (Reference Data not found).

Required Vendor Information

VE05.22.01B: The vendor shall state in its documentation that the card supports the assertion made in AS05.22B.

Required Test Procedures

TE05.22.01B: The tester shall validate that the vendor documentation contains the information required in VE05.22.01B and the card returns status word ‘6A 88’, when the key reference is set to a value other than what is supported by the card.

A.5.2.2 CHANGE REFERENCE DATA Card Command

AS05.23 and AS05.24: Only reference data associated with key references ‘80’ and ‘81’ specific to the PIV Card Application (i.e., local key reference) and the Global PIN with key reference ‘00’ may be changed by the PIV card Application Change Reference Data command. Key reference ‘80’ reference data shall be changed by the PIV Card Application CHANGE REFERENCE DATA command. The ability to change reference data associated with key references '81' and '00' using the PIV Card Application CHANGE REFERENCE DATA command is optional.

Required Vendor Information

VE05.23 and VE05.24: The vendor shall state in its documentation that the card supports the assertion made in AS05.023 and AS05.24.

Required Test Procedures

TE05.23.01 and TE05.24.01: The tester shall validate that reference data associated with key reference '80' can be changed by the PIV Card Application's CHANGE REFERENCE DATA command. If the Discovery Object is implemented with PIN Usage Policy first byte set to 0x60, and the implementation supports changing the Global PIN with the CHANGE REFERENCE DATA command, then the tester shall also validate that key reference '00' reference data can be changed by the CHANGE REFERENCE DATA command.

AS05.25: If the current value of the retry counter associated with the key reference is zero, then the reference data associated with the key reference shall not be changed and the PIV Card Application shall return the status word '69 83' (Authentication method blocked).

Required Vendor Information

VE05.25.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.016.

Required Test Procedures

TE05.25.01: The tester shall validate that when the current value of the retry counter associated with the key reference is zero, the reference data associated with the key reference does not change and the PIV Card Application returns '69 83' (Authentication method blocked).

AS05.26 If the card command succeeds, then the security status of the key reference shall be set to TRUE and the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference.

Required Vendor Information

VE05.26.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.26.

Required Test Procedures

TE05.26.01: The tester shall validate that the vendor documentation states the required information in VE05.26.01 and the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference when the command succeeds.

AS05.27: If the card command fails, then the security status of the key reference shall be set to FALSE and the retry counter associated with the key reference shall be decremented by one.

Required Vendor Information

VE05.27.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.18.

Required Test Procedures

TE05.27.01: The tester shall validate that the vendor documentation contains the information required in VE05.27.01 and the retry counter associated with the key reference shall be decremented by one if the card command fails.

AS05.28: If the either the current reference data or the new reference data in the command data field of the command does not satisfy the criteria in Section 2.4.3, Part 2 of SP 800-73-3, the PIV Card Application shall not change the reference data associated with the key reference and shall return the status word '6A 80'.

Required Vendor Information

VE05.28.01: The vendor shall state in its documentation that the card supports the assertion made in AS05. 28.

Required Test Procedures

TE05.28.01: The tester shall validate that 1) the vendor documentation contains the information required in VE05.19.01, 2) the old PIN is not changed, and 3) the card returns status word ‘6A 80’ when the PIN information in the reference data field of the command is not padded to 8 bytes.

AS05.28A: If the key reference is set to a value other than what is supported by the card, the PIV Card Application … shall return the status word ‘6A 88’ (Key Reference not found).

Required Vendor Information

VE05.28.01.A: The vendor shall state in its documentation that the card supports the assertion made in AS05.28A.

Required Test Procedures

TE05.28.01A: The tester shall validate that the vendor documentation contains the information required in VE05.28.01A, the old PIN is not changed, and the card returns status word ‘6A 88’, when the key reference is set to a value other than what is supported by the card.

A.5.2.3 RESET RETRY COUNTER Card Command

AS05.29: The only key reference allowed in the P1 parameter of the RESET RETRY COUNTER command is the PIV Card Application PIN.

.

Note: This assertion is not tested separately.

AS05.30: If the current value of the PUK's retry counter is zero then the PIN's retry counter shall not be reset and the PIV Card Application shall return the status word '69 83' (Authentication method blocked).

Required Vendor Information

VE05.30.01: same as VE05.17.01.

VE05.30.02: The vendor shall specify in its documentation that the RESET RETRY COUNTER card command will not reset the retry counter and the card will return '69 83' (Authentication method blocked) when the PUK's retry counter associated with the key reference is zero.

Required Test Procedures

TE05.30.01: The tester shall validate that the information requested in VE05.30.02 and VE05.30.01 are present in the vendor documentation. (NOTE: Testing this condition will leave the card unusable for further tests since the reset counter is zero).

AS05.31: If the card command succeeds, then the PIN's retry counter shall be set to its reset retry value. Optionally, the PUK's retry counter may be set to its initial reset retry value. The security status of the PIN's key reference shall not be changed.

Required Vendor Information

VE05.31.01: same as VE05.17.01

VE05.31.02: The vendor shall specify in its documentation that the card supports the assertion made in AS05.31.

Required Test Procedures

TE05.31.01: The tester shall validate that when the card command succeeds, the PIN's retry counter is set to the PIN's reset retry value specified in VE05.31.01, and the security status of the PIN's key reference is not changed. If the PUK's retry counter can be reset, the tester shall validate that the PUK's retry counter was reset to its initial reset retry value.

AS05.32: If the card command fails, then the security status of the PIN's key reference shall be set to FALSE and the PUK's retry counter shall be decremented by one.

Required Vendor Information

VE05.32.01: The vendor shall state in its documentation that card supports the assertion made in AS05.032.

Required Test Procedures

TE05.32.01: The tester shall validate that the information requested in VE05.32.01 is present in the vendor documentation, the security status of the PIN's key reference is set to FALSE and the PUK's retry counter is decremented by one.

AS05.33: If the either the reset retry counter reference data (PUK) or the new reference data (PIN) in the command field of the command does not satisfy the criteria in Section 2.4.3, Part 2 of SP 800-73-3, the PIV Card Application shall not reset the retry counter associated with the PIN and shall return the status word '6A 80'.

Required Vendor Information

VE05.33.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.33.

Required Test Procedures

TE05.33.01: The tester shall validate that the vendor documentation includes the information required in VE05.24.01 and that when either the PUK or the PIN of the command does not satisfy the criteria in Section 2.4.3 Part 2 of SP 800-73-3, the PIN's retry counter is not reset and the card returns '6A 80.'

AS05.33A: If the key reference value is other than what is supported by the card, the PIV Card Application shall … return the status word '6A 88' (Key Reference not found).

Required Vendor Information

VE05.33.01A: The vendor shall state in its documentation that the card supports the assertion made in AS05.33A.

Required Test Procedures

TE05.33.01A: The tester shall validate that the vendor documentation includes the information required in VE05.33.01A and that when the key reference value is other than what is supported by the card, the card returns '6A 88'.

A.5.2.4 GENERAL AUTHENTICATE Card Command

AS05.34: The GENERAL AUTHENTICATE card command performs a cryptographic operation such as an authentication protocol using the data provided in the data field of the command and returns the result of the cryptographic operation in the response data field.

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

2) **The GENERAL AUTHENTICATE command shall be used with the PIV Digital Signature Key (‘9C’) to realize the signing functionality on the PIV client application programming interface. Data to be signed is expected to be hashed off-card.**

3) **The GENERAL AUTHENTICATE command shall be used with the PIV Key Management Key ('9D') and the retired PIV Key Management Keys ('82' – '95') to realize key establishment schemes specified in SP 800-78 (ECDH and RSA).**

Required Vendor Information

VE05.34.01: The vendor shall specify in its documentation the types of cryptographic operations (authentication, key establishment, signing) supported by the card.

Required Test Procedures

TE05.34.01: The tester shall validate that the GENERAL AUTHENTICATE command is implemented to authenticate the Card to the client application.

TE05.34.02: The tester shall validate that the GENERAL AUTHENTICATE command is implemented to authenticate the client application to the card.

TE05.34.03: The tester shall validate that the GENERAL AUTHENTICATE command is implemented to mutually authenticate the Card to the client application and the client application to the card.

TE05.34.04: If the 9C key is implemented, the tester shall validate that the GENERAL AUTHENTICATE command is implemented to realize signing functionality.

TE05.34.05: If the 9D key is implemented, the tester shall validate that the GENERAL AUTHENTICATE command is implemented to support the RSA key transport or Elliptic Curve Diffie Hellman key establishment schemes specified in SP800-78.

AS05.35: The GENERAL AUTHENTICATE command shall be implemented to realize the signing functionality on the PIV client application programming interface.

AS05.35 is satisfied by AS05:34 and its TE05.34.04.

AS05.36: If an invalid value of algorithm reference (P1) and/or key reference (P2) is sent to the card, the PIV Card Application shall return the status word '6A 86'.

Required Vendor Information

VE05.36.01: The vendor shall state in its documentation that the card supports the assertion made in AS05.36.

Required Test Procedures

TE05.36.01: The tester shall validate that the vendor documentation contains the information required in VE05.36.01 and the card returns status word ‘6A 86’, when an invalid value of algorithm reference (P1) or key reference (P2) is sent to the card.

AS05.36A: If an invalid value is sent in the data field, the PIV Card Application shall return the status word '6A 80'.

Required Vendor Information

VE05.36.01A: The vendor shall state in its documentation that the card supports the assertion made in AS05.36A.

Required Test Procedures

TE05.36.01A: The tester shall validate that the vendor documentation contains the information required in VE05.36.01A and the card returns status word ‘6A 80’ when an invalid value in data field of the command is sent to the card.

AS05.36B: If the command is used to authenticate the Card to the client application using a PIN-protected PIV key, without prior PIN verification, the PIV Card Application shall return the status word '69 82'.

Required Vendor Information

VE05.36.01B: The vendor shall state in its documentation that the card supports the assertion made in AS05.36B.

Required Test Procedures

TE05.36.01B: The tester shall validate that the vendor documentation contains the information required in VE05.36.01B and the card returns status word ’69 82 whenever the command is used to authenticate the card to the client application using a PIN protected key without prior PIN verification.

AS05.36C: If a card command other than the GENERAL AUTHENTICATE command is received by the PIV Card Application before the termination of a GENERAL

AUTHENTICATE chain, the PIV Card Application shall rollback to the state it was in immediately prior to the reception of the first command in the interrupted chain.

Required Vendor Information

VE05.36.01C: The vendor shall specify in its documentation that the card supports the assertion made in AS05.36C.

Required Test Procedures

TE05.36.01C: The tester shall validate by inspection of the vendor documentation that the PIV Card Application reverts back to the state it was in if a command other than GENERAL AUTHENTICATE is received before the termination of a GENERAL AUTHENTICATE chain.

A.5.3 PIV Card Application Card Commands for Credential Initialization and Administration

A.5.3.1 PUT DATA Card Command

AS05.37: The PUT DATA card command completely replaces the data content of a single data object in the PIV Card Application with new content.

Required Vendor Information

VE05.37.01: The vendor shall specify in its documentation the format, encoding, and the parameters of the PUT DATA command supported by the card.

Required Test Procedures

TE05.37.01: The tester shall validate that the card complies with the PUT DATA command as defined in SP 800-73-3, Part 2.

A.5.3.2 GENERATE ASYMMETRIC KEY PAIR Card Command

AS05.38 The GENERATE ASYMMETRIC KEY PAIR card command initiates the generation and storing in the card of the reference data of an asymmetric key pair, i.e., a public key and a private key, and the command returns the public key.

Required Vendor Information

VE05.38.01: The vendor shall specify in its documentation the cryptographic mechanism identifiers (specified in table 4, Part 1 of SP 800-73-3) that have been implemented on the card.

Required Test Procedures

TE05.38.01: The tester shall validate that the card implements the algorithms associated with identifiers specified as part of VE05.38.01 requirement and that the public key returned is formatted based on data object tags specified in Table 9, Part 2 of SP 800-73-3.

AS05.39: The public key of the generated key pair is returned as the response to the command.

Note: This assertion is tested as part of AS05.38.

AS05.40: If there is reference data currently associated with the key reference, it is replaced in full by the generated data.

Required Vendor Information

VE05.40.01: The vendor shall provide the contents of the public key data on the card.

Required Test Procedures

TE05.40.01: The tester shall validate that the initial contents of the public key data is replaced in full by the generated data, following a GENERATE ASYMMETRIC KEY PAIR command.

Appendix B—PIV Client API Test Assertions

Test Assertion Template

Purpose	A quick description of the test and why it is being run
Target	The PIV function call being tested
Reference(s)	References to the SP 800-73-3 or other relevant publications
Precondition(s)	Anything that must be done or known prior to executing the scenario
Test Steps	`Sequence of steps for making a function call`
Expected Result(s)	What the expected execution path yields in terms of progress and values
Post Condition(s)	A description of both client and card application state once the test scenario completes

B.1 Connection Test Assertions

B.1.1 Valid Path Test Assertions

B.1.1.1 Initiate Exclusive Connection

Purpose	Confirms that an exclusive connection can be obtained by a calling application to the PIV Card Application on a specific ICC
Target	`pivConnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.01, AS04.02
Precondition(s)	1. A valid connection description is provided for the card application 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application 3. No application is currently connected to the PIV Card Application
Test Steps	1. Set sharedConnection := false 2. Set connectionDescription := <<valid connection>> 3. Create cardHandle reference 4. Call pivConnect w/ • *(IN)* sharedConnection • *(INOUT)* connectionDescription • *(INOUT) CDLength* • *(OUT)* cardHandle
Expected Result(s)	Call returns with status_word := PIV_OK and initialized cardHandle
Post Condition(s)	Client Application is connected to PIV Card

B.1.1.2 Initiate Shared Connection

Purpose	Confirms that a shared connection can be established by two distinct calling applications to the PIV Card with a specific ICC
Target	`pivConnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.02
Precondition(s)	1. A valid connection description is provided for the card application 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application 3. Another client application is currently connected via a shared connection to the PIV Card Application.
Test Steps	1. Set sharedConnection := true 2. Set connectionDescription := <<valid connection>> 3. Create cardHandle reference 4. Call pivConnect w/ • *(IN)* sharedConnection • *(INOUT)* connectionDescription • *(INOUT) CDLength* • *(OUT)* cardHandle
Expected Result(s)	Call returns with status_word := PIV_OK and initialized cardHandle
Post Condition(s)	Both client applications are connected through the same connection to the PIV Card Application.

B.1.2 Test Assertions for Error Conditions

B.1.2.1 Malformed Connection Description

Purpose	Confirms that the correct status word is returned when a malformed connection description is used
Target	`pivConnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.02
Precondition(s)	1. An invalid connection description is provided for the card application 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application
Test Steps	1. Set sharedConnection := true | false 2. Set connectionDescription := <<invalid connection>> 3. Create cardHandle reference 4. Call pivConnect w/ • *(IN)* sharedConnection • *(INOUT)* connectionDescription • *(INOUT) CDLength* • *(OUT)* cardHandle
Expected Result(s)	Call returns with status_word PIV_CONNECTION_DESCRIPTION_MALFORMED

Post Condition(s)	1. The cardHandle variable is not initialized 2. The Client Application is not connected to the PIV Card Application

B.1.2.2 Attempting to Share/Lock an Exclusive Connection

Purpose	Ensure that when an unshared connection is initially established that no additional connections can be established
Target	`pivConnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.02
Precondition(s)	1. A valid connection description is provided for the card application 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application 3. An application currently owns an exclusive connection (`sharedConnection := false`)

Test Steps	`1. Set sharedConnection := true | false`
	`2. Set connectionDescription := <<valid connection>>`
	`3. Create cardHandle reference`
	`4. Call pivConnect w/`
	• *(IN)* `sharedConnection`
	• *(INOUT)* `connectionDescription`
	• *(INOUT) CDLength*
	• *(OUT)* `cardHandle`

Expected Result(s)	Call returns with status_word := `PIV_CONNECTION_LOCKED`
Post Condition(s)	1. The Client Application previously connected remains connected 2. The cardHandle variable is not initialized 3. The newly requesting Client Application is not connected to the PIV Card Application

B.1.2.3 Attempting to Lock a Shared Connection

Purpose	Ensure that a Client Application cannot lock a PIV Card application connection that currently has open shared connections
Target	`pivConnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.02
Precondition(s)	1. A valid connection description is provided for the card application 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application 3. An application currently owns a shared connection (`sharedConnection := true`)
Test Steps	`1. Set sharedConnection := false`

	2. Set connectionDescription := <<valid connection>> 3. Create cardHandle reference 4. Call pivConnect w/ • *(IN)*sharedConnection • *(INOUT)* connectionDescription • *(INOUT) CDLength* • *(OUT)* cardHandle
Expected Result(s)	Call returns with status_word := PIV_CONNECTION_FAILURE
Post Condition(s)	1. The Client Application previously connected remains connected 2. The cardHandle variable is not initialized 3. The newly requesting Client Application is not connected to the PIV Card Application

B.1.2.4 Attempting to Open an Unsupported Connection

Purpose	Confirms that the correct status word is returned when an unsupported connection mode is attempted.
Target	pivConnect
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.2 2. AS03.04, AS04.02
Precondition(s)	1. An invalid connection mode (e.g. Integrated Services Digital Network (ISDN)) is attempted 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application
Test Steps	1. Set sharedConnection := true | false 2. Set connectionDescription := <<valid ISDN connection string>> 3. Create cardHandle reference 4. Call pivConnect w/ • *(IN)* sharedConnection • *(INOUT)* connectionDescription • *(INOUT) CDLength* • *(OUT)* cardHandle
Expected Result(s)	Call returns with status_word := PIV_CONNECTION_FAILURE
Post Condition(s)	1. The cardHandle variable is not initialized 2. The Client Application is not connected to the PIV Card

B.2 Disconnection Test Assertions

B.2.1 Valid Test Assertions

B.2.1.1 Disconnect an Exclusive Connection

Purpose	Ensure that a Client Application can close a currently open exclusive PIV Card application connection
Target	pivDisconnect
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.3

	2. AS03.04, AS04.01, AS04.03
Precondition(s)	1. There exists a valid physical and logical connection between an instance of the PIV Card and the host of the calling application 2. A client application currently has a connection accessible through `cardHandle`
Test Steps	`1. Call pivDisconnect w/ arguments` • *`(IN)`* `cardHandle`
Expected Result(s)	Call returns with status_word := `PIV_OK`
Post Condition(s)	1. The client application is no longer connected to the PIV card application 2. PIV Card Application is no longer aware of the Client Application

B.2.1.2 Disconnect a Shared Connection

Purpose	Ensure that a Client Application can close a currently open and shared PIV Card Application connection without impacting other Client Application's connections to that same PIV Card Application
Target	`pivDisconnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.3 2. AS04.03
Precondition(s)	1. There exists a valid logical and physical connection between an instance of the PIV Card and the host of the calling application 2. At least two distinct client applications (having two distinct `CardHandle references`) are connected to the PIV Card Application
Test Steps	`1. Call pivDisconnect w/ arguments` • *`(IN)`* `cardHandle`
Expected Result(s)	Call returns with status_word := `PIV_OK`
Post Condition(s)	1. The Client Application is no longer connected to the PIV Card Application 2. All other Client Applications maintain their previously valid connections 3. PIV Card Application is no longer aware of that particular Client Application but remains aware of all other Client Applications.

B.2.2 Test Assertions for Error Cases

B.2.2.1 Attempt Disconnect with Invalid Card Handle

Purpose	Ensure that the Client Application can detect an invalid cardHandle argument.
Target	`pivDisconnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.3

	2. AS04.03
Precondition(s)	1. There exists a valid physical and logical connection between an instance of the PIV Card and the host of the calling application 2. A client application currently has a connection accessible through `cardHandle`

Test Steps	`1. Set cardHandle := <<invalid cardHandle>>`
	`2. Call pivDisconnect w/`
	• *`(IN)`* `cardHandle`

Expected Result(s)	Call returns with status_word := `PIV_INVALID_CARD_HANDLE`
Post Condition(s)	The client application remains connected to the PIV Card Application

B.2.2.2 Disconnecting a previously disconnected Client Application

Purpose	Verify that if a Client Application tries to close a previously closed PIV Card Application connection (i.e., with the same cardHandle), the application returns an Invalid Card Handle message.
Target	`pivDisconnect`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.1.3 2. AS04.03
Precondition(s)	1. There exists a client application with a valid and open `cardHandle:CardHandle` to a PIV Card Application 2. The subject connection was previously closed 3. The card is physically connected to the card reader
Test Steps	`1. Call pivDisconnect w/ arguments` • *`(IN)`* `cardHandle`
Expected Result(s)	Call returns with status_word := `PIV_INVALID_CARD_HANDLE`
Post Condition(s)	1. The Client Application remains unconnected to the PIV Card Application 2. PIV Card Application remains unaware of that particular Client Application

B.3 pivSelectCardApplication

B.3.1 Valid Test Assertions

B.3.1.1 Select a Card Application with a full AID

Purpose	Ensure that a Client Application can locate and select a valid Card Application, store its properties, and return a reference to the Application Properties.
Target	`pivSelectCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.1 2. AS03.04, AS04.01, AS04.04

Precondition(s)	1. The Client Application currently owns a connection accessible through cardHandle.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set applicationID := <<valid applicationID>> 3. Create applicationProperties reference 4. Call pivSelectCardApplication w/ • *(IN)* cardHandle • *(IN)* applicationAID • *(IN) aidLength* • *(OUT)* applicationProperties • *(INOUT) APLength*
Expected Result(s)	Call returns with status_word of PIV_OK and initialized applicationProperties reference
Post Condition(s)	The "CurrentlySelectedApplication" of the PIV Card is the PIV Card Application. The PIV Card Application's security state is established.

B.3.1.2 Use a right truncated AID to Select a Card Application

Purpose	Ensure that a Client Application is able to locate and select a valid Card Application that is identified by a right truncated AID, store its properties, and return a reference to the applicationProperties.
Target	pivSelectCardApplication
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.1 2. AS04.04
Precondition(s)	1. The Client Application currently owns a connection accessible through cardHandle.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set applicationID := <<valid right truncated applicationID>> 3. Create applicationProperties reference 4. Call pivSelectCardApplication w/ • *(IN)* cardHandle • *(IN)* applicationAID • *(IN) aidLength* • *(OUT)* applicationProperties • *(INOUT) APLength*
Expected Result(s)	1. Call returns with status_word of PIV_OK and initialized applicationProperties reference
Post Condition(s)	The "CurrentlySelectedApplication" in PIV Card is the PIV Card Application. The PIV Card Application security state is established.

B.3.2 Test Assertions for Error Conditions

B.3.2.1 Detect and handle an invalid cardHandle reference

Purpose	Ensure that a Client Application can detect and gracefully exit when passed an invalid cardHandle.

Target	pivSelectCardApplication
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.1 2. AS04.04
Precondition(s)	1. An invalid cardHandle is passed to the client application. 2. The applicationAID is assumed to be valid.
Test Steps	1. Set cardHandle := <<invalid cardHandle>> 2. Set applicationID := <<valid applicationID>> 3. Create applicationProperties reference 4. Call pivSelectCardApplication w/ • *(IN)* cardHandle • *(IN)* applicationAID • *(IN) aidLength* • *(OUT)* applicationProperties • *(INOUT) APLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_CARD_HANDLE and does not initialize applicationProperties reference
Post Condition(s)	The Client Application returns to the state it had prior to calling pivSelectCardApplication.

B.3.2.2 Detect and handle an invalid applicationAID

Purpose	Ensure that a Client Application can detect and gracefully exit when passed an invalid applicationAID.
Target	pivSelectCardApplication
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.1 2. AS04.04
Precondition(s)	1. A correctly formatted but invalid applicationAID is passed to the client application. 2. The cardHandle is assumed to be valid.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set applicationID := <<invalid applicationID>> 3. Create applicationProperties reference 4. Call pivSelectCardApplication w/ • *(IN)* cardHandle • *(IN)* applicationAID • *(IN) aidLength* • *(OUT)* applicationProperties • *(INOUT) APLength*
Expected Result(s)	Call returns with status_word of PIV_CARD_APPLICATION_NOT_FOUND and does not initialize applicationProperties reference
Post Condition(s)	The Client Application returns to the state it had prior to calling pivSelectCardApplication.

B.4 pivLogIntoCardApplication

B.4.1 Valid Test Assertions

B.4.1.1 Log on to the Card Application

Purpose	Validate that the Client Application can set (update) application security status with the selected PIV Card Application.
Target	`pivLogIntoCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.2 2. AS03.04, AS04.01, AS04.05
Precondition(s)	1. The card has established a connection to the client. 2. The cardHandle was properly initialized by pivConnect. 3. The client application has successfully executed the pivSelectCardApplication command.

Test Steps	`1. Set cardHandle := <<a valid cardHandle>>`
	`2. Set authenticators := <<valid authenticators byte sequence>>`
	`3. Call pivLogIntoCardApplication w/`
	• *(IN)* `authenticators`
	• *(IN) AuthLength*
	• *(IN)* `cardHandle`

Expected Result(s)	Call returns with status_word of `PIV_OK`
Post Condition(s)	Security context is set (updated) and the Client Application can now perform read operations on PIN-protected data objects controlled by the PIV Card Application. The client is thus logged into the PIV Card Application.

B.4.2 Test Assertions for Error Conditions

B.4.2.1 Attempt Logon with an invalid cardHandle

Purpose	Ensure a Client Application can detect and process an invalid card handle.
Target	`pivLogIntoCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.2 2. AS04.05
Precondition(s)	1. The card has established a connection to the client. 2. The cardHandle was properly initialized by pivConnect. 3. The client application has successfully executed the pivSelectCardApplication command.

Test Steps	`1. Set cardHandle := <<an invalid cardHandle>>`
	`2. Set authenticators := <<valid authenticators byte sequence>>`
	`3. Call pivLogIntoCardApplication w/`
	• *(IN)* `authenticators`
	• *(IN) AuthLength*
	• *(IN)* `cardHandle`

Expected Result(s)	Call returns with status_word := `PIV_INVALID_CARD_HANDLE`
Post Condition(s)	The Client Application is not logged into the Card Application. The

	client application was not able to set (update) the application security status with the selected PIV Card Application

B.4.2.2 Attempt Logon with a malformed authenticator

Purpose	Ensure a Client Application can detect and process a malformed authenticator byte sequence.
Target	`pivLogIntoCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.2 2. AS04.05
Precondition(s)	1. The card has established a connection to the client. 2. The cardHandle was properly initialized by pivConnect. 3. The client application has successfully executed the pivSelectCardApplication command.
Test Steps	1. Set cardHandle := <<a valid cardHandle>> 2. Set authenticators := <<a malformed authenticators byte sequence>> 3. Call pivLogIntoCardApplication w/ • *(IN)* authenticators • *(IN) AuthLength* • *(IN)* cardHandle
Expected Result(s)	Call returns with status_word := `PIV_AUTHENTICATOR_MALFORMED`
Post Condition(s)	The Client Application is not logged into the Card Application. The client application was not able to set (update) the application security status with the selected PIV Card Application.

B.4.2.3 Attempt Logon with invalid authenticator

Purpose	Ensure a Client Application can detect and process an authenticator that has the correct format but does not result in a valid security permission/context.
Target	`pivLogIntoCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.2 2. AS04.05
Precondition(s)	1. The card has established a connection to the client. 2. The cardHandle was properly initialized by pivConnect. 3. The client application has successfully executed the pivSelectCardApplication command.
Test Steps	1. Set cardHandle := <<a valid cardHandle>> 2. Set authenticators := <<a well formed authenticators byte sequence containing an invalid PIN and/or KEY_REFERENCE value>> 3. Call pivLogIntoCardApplication w/ • *(IN)* authenticators • *(IN) AuthLength* • *(IN)* cardHandle

Expected Result(s)	Call returns with status_word := `PIV_AUTHENTICATION_FAILURE`
Post Condition(s)	The Client Application is not logged into the Card Application. The client application was not able to set (update) the application security status with the selected PIV Card Application.

B.5 pivLogoutOfCardApplication

B.5.1 Valid Test Assertions

B.5.1.1 Log out of the Card Application

Purpose	Reset security context of the card application.
Target	`pivLogoutOfCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.4 2. AS03.04, AS04.01, AS04.07
Precondition(s)	1. The client has established a connection to the card. 2. The client is logged into the card application. 3. The client has established an "application security status".
Test Steps	`1. Set cardHandle := <<a valid cardHandle>>` `2. Call pivLogoutOfCardApplication w/ (IN)cardHandle`
Expected Result(s)	Call returns with status_word := `PIV_OK` and the Client Application is logged off of the Card Application
Post Condition(s)	1. The Client Application is logged off of the Card Application. Only "free read" data can be read. 2. The cardHandle remains valid. 3. The connection remains open.

B.5.1.2 Attempt Log out without logging in

Purpose	Verify that logging out without logging does not return any error condition.
Target	`pivLogoutOfCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.4 2. AS04.07
Precondition(s)	1. The client has established a connection to the card. 2. The client has successfully executed the pivSelectCardApplication method. 3. The client is not logged into the PIV Card Application
Test Steps	`1. Set cardHandle := <<a valid cardHandle>>` `2. Call pivLogoutOfCardApplication w/(IN)cardHandle`
Expected Result(s)	Call returns with status_word := `PIV_OK`
Post Condition(s)	The precondition states remain unchanged. (Only "free read" data can be read)

B.5.2 Test Assertions for Error Conditions

B.5.2.1 Attempt Log out with Invalid Cardhandle

Purpose	Ensure the method can detect and handle an invalid cardHandle.
Target	`pivLogoutOfCardApplication`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.4 2. AS04.07
Precondition(s)	1. The client has established a connection to the card. 2. The client is logged into the card application. 3. The client has established an "application security status".
Test Steps	1. Set cardHandle := <<an invalid cardHandle>> 2. Call `pivLogoutOfCardApplication` w/*(IN)*cardHandle
Expected Result(s)	Call returns with status_word :=`PIV_INVALID_CARD_HANDLE`
Post Condition(s)	The precondition states remain unchanged.

B.6 pivGetData

B.6.1 Valid Test Assertions

B.6.1.1 Get a reference to data object that does not require Login

Purpose	Ensure the Client Application can read data objects from the card that does not require a Login.
Target	`pivGetData`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS03.04, AS04.01, AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is not logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID>> (Repeat this for all implemented objects on the card except for Fingerprint, Printed Information, Facial Image and Iris Image) 3. Create data reference 4. Call `pivGetData` w/ (each data object identified in Step 2) • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word of `PIV_OK in each case` and an initialized reference to data
Post Condition(s)	N/A

B.6.1.2 Get a reference to data object that requires Login

Purpose	Ensure the Client Application can read data objects from the card that requires a Login.

Target	pivGetData
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS03.04, AS04.01, AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID>> (Repeat this for all implemented objects in the following set - Fingerprint, Printed Information, Facial Image and Iris Image) 3. Create data reference 4. Call pivGetData w/ (each data object identified in step 2) • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word of PIV_OK in all cases and an initialized reference to data
Post Condition(s)	N/A

B.6.2 Test Assertions for Error Conditions

B.6.2.1 Handle an invalid cardHandle

Purpose	Ensure the Client Application can recognize and handle an invalid cardHandle.
Target	pivGetData
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application is not logged into the Card Application.
Test Steps	1. Set cardHandle := <<invalid cardHandle>> 2. Set OID := <<valid OID>> 3. Create data reference 4. Call pivGetData w/ • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word := PIV_INVALID_CARD_HANDLE and does not initialize data reference
Post Condition(s)	The client application returns to the state it had before the call.

B.6.2.2 Handle an invalid Object Identifier

Purpose	Ensure the Client Application can recognize and handle an invalid OID.
Target	`pivGetData`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application is logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<invalid OID>> (Improper syntax or not found in Table 2 of SP 800-73-3 Part 1) 3. Create data reference 4. Call pivGetData w/ • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word := `PIV_INVALID_OID` and does not initialize data reference
Post Condition(s)	The client application returns to the state it had before the call.

B.6.2.3 The Client Application can handle missing data object

Purpose	Ensure the Client Application can recognize and handle a missing or unrecognized OID.
Target	`pivGetData`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID>> (Found in Table 2 of SP 800-73-3 Part 1 but not implemented on the PIV Card application) NOTE: A valid OID not found in table 2 can be used instead, if the card application has implemented all optional data object. 3. Create data reference 4. Call pivGetData w/ • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word := `PIV_DATA_OBJECT_NOT_FOUND` and

	does not initialize data reference (NOTE: This test will return PIV_INVALID_OID if the card has implemented all optional objects in the PIV Data Model)
Post Condition(s)	The client application returns to the state it had before the call.

B.6.2.4 Security Conditions are enforced for Secured Objects

Purpose	Ensure that Security Conditions are enforced for Retrieving Data from Secured Applications
Target	`pivGetData`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.2.3 2. AS04.06
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client has successfully selected the PIV Card Application 3. The client is not logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID for each of the following objects; Fingerprints, Facial Image, Printed Information and Iris Image >> 3. Create data reference 4. Call pivGetData w/ (each data object identified in step 2) • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(OUT)* data • *(INOUT) DataLength*
Expected Result(s)	Call returns with status_word := PIV_SECURITY_CONDITION_NOT_SATISFIED and does not initialize data reference
Post Condition(s)	The client application returns to the state it had before the call.

B.7 pivPutData

B.7.1 Valid Test Assertions

B.7.1.1 Write data to an object on the Card through the Client Application

Purpose	Ensure the Client Application can write the entire data content to an object on the Card Application.
Target	`pivPutData`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.1 2. AS03.04, AS04.01, AS04.09
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle.

	2. Authentication between the PIV Card Application and the PIV Card Application Administrator has taken place.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID>> 3. Set data := <<a correctly formatted byte sequence> 4. Call pivPutData w/ (for all data objects) • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(IN)* data • *(IN) dataLength*
Expected Result(s)	Call returns with status_word of PIV_OK for each test case.
Post Condition(s)	Validate that the Card Application has written the entire dataset of the selected object on the Client Application by issuing pivGetData

B.7.2 Test Assertions for Error Conditions

B.7.2.1 Identify and handle an invalid card handle

Purpose	Ensure the Client Application can identify and respond to an invalid card handle.
Target	pivPutData
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.1 2. AS04.09
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The PIV Card Application has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<invalid cardHandle>> 2. Set OID := <<valid OID>> 3. Set data := <<a correctly formatted byte sequence> 4. Call pivPutData w/ • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(IN)* data • *(IN) dataLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_CARD_HANDLE
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivPutData method call. 2. The precondition states remain unchanged.

B.7.2.2 Identify and handle an invalid Object Identifier (OID)

Purpose	Ensure the Client Application can identify and handle an invalid OID.

Target	pivPutData
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.1 2. AS04.09
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The PIV Card has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<invalid OID>> (Improper syntax or not found in Table 2 of SP 800-73-3 Part 1) 3. Set data := <<a correctly formatted byte sequence> 4. Call pivPutData w/ • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(IN)* data • *(IN) dataLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_OID.
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivPutData function call. 2. The precondition states remain unchanged.

B.7.2.3 Security Conditions are enforced for writing data to the on-card data containers

Purpose	Ensure that Security Conditions are enforced for writing data to the PIV Card Application
Target	pivPutData
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.1 2. AS04.09
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has not authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set OID := <<valid OID>> 3. Create data reference 4. Call pivPutData w/ (for all data objects) • *(IN)* cardHandle • *(IN)* OID • *(IN) oidLength* • *(IN)* data • *(IN) dataLength*
Expected Result(s)	All calls return with status_word :=

	PIV_SECURITY_CONDITION_NOT_SATISFIED and does not initialize data reference
Post Condition(s)	The client application returns to the state it had before the call.

B.8 pivGenerateKeyPair

B.8.1 Valid Test Assertions

B.8.1.1 Generate an asymmetric key pair

Purpose	Ensure the Card Application can generate an asymmetric key pair.
Target	pivGenerateKeyPair
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.2 2. AS03.04, AS04.01, AS04.10
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<an existing key reference suitable for use with the specified cryptographicMechanism >> (say 9A) 3. Set cryptographicMechanism := <<the recognized Cryptographic Mechanism Identifier for the key reference value '9A' are: <<06, 07 or 11>>. See table 6-3 in SP 800-78-2 4. Create publicKey reference 5. Call pivGenerateKeyPair w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* cryptographicMechanism • *(OUT)* publicKey • *(INOUT) KeyLength* *6: Repeat steps 1 through 5 for each on-card generated asymmetric PIV key type.*
Expected Result(s)	Call returns with status_word of PIV_OK and a reference to publicKey
Post Condition(s)	The Client Application creates a reference to a public key / private key pair which is accessible to the Card Application.

B.8.2 Test Assertions for Error Conditions

B.8.2.1 Identify and handle an invalid card handle

Purpose	Ensure the Card Application can catch invalid card handles.
Target	pivGenerateKeyPair

Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.2 2. AS04.10
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<invalid cardHandle>> 2. Set keyReference := <<an existing key reference suitable for use with the specified cryptographicMechanism >> 3. Set cryptographicMechanism := <<a recognized Cryptographic Mechanism Identifier>> 4. Create publicKey reference 5. Call pivGenerateKeyPair w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* cryptographicMechanism • *(OUT)* publicKey • *(INOUT) KeyLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_CARD_HANDLE
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivGenerateKeyPair function call. 2. The precondition states are unaffected.

B.8.2.2 Identify and handle an invalid keyReference or key-Algorithm Combination

Purpose	Ensure that the Card Application can identify an invalid keyReference.
Target	pivGenerateKeyPair
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.2 2. AS04.10
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<a key reference not found in the specification>> 3. Set cryptographicMechanism := <<a recognized Cryptographic Mechanism Identifier>> 4. Create publicKey reference 5. Call pivGenerateKeyPair w/ • *(IN)* cardHandle

	• *(IN)* keyReference • *(IN)* cryptographicMechanism • *(OUT)* publicKey • *(INOUT) KeyLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_KEY_OR_KEYALG_COMBINATION
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivGenerateKeyPair method call. 2. The precondition states are unaffected.

B.8.2.3 Identify and handle an invalid cryptographicMechanism

Purpose	Ensure that the Card Application can identify unsupported cryptographicMechanisms.
Target	pivGenerateKeyPair
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.2 2. AS04.10
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<a valid key reference>> 3. Set cryptographicMechanism := <<an unrecognized Cryptographic Mechanism Identifier>> 4. Create publicKey reference 5. Call pivGenerateKeyPair w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* cryptographicMechanism • *(OUT)* publicKey • *(INOUT) KeyLength*
Expected Result(s)	Call returns with status_word of PIV_UNSUPPORTED_CRYPTOGRAPHIC_MECHANISM
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivGenerateKeyPair function call. 2. The precondition states are unaffected.

B.8.2.4 Security Conditions are Enforced

Purpose	Ensure that the card application enforces the necessary security conditions when called from Client Application.
Target	pivGenerateKeyPair
Reference(s)	1. SP 800-73-3 Part 3, Section 3.4.2

	2. AS04.10
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The Client Application has successfully selected the PIV Card Application. 3. The PIV Card Application has not authenticated the PIV Card Application Administrator.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set cryptographicMechanism := <<a recognized Cryptographic Mechanism Identifier>> 3. Set keyReference := <<a reference to a valid key that is associated with the selected cryptographicMechanism >> 4. Create publicKey reference 5. Call pivGenerateKeyPair w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* cryptographicMechanism • *(OUT)* publicKey • *(INOUT) KeyLength*
Expected Result(s)	Call returns with status_word of PIV_SECURITY_CONDITIONS_NOT_SATISFIED
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivGenerateKeyPair method call. 2. The precondition states are unaffected.

B.9 pivCrypt

B.9.1 Valid Test Assertions

B.9.1.1 Authenticate the Card Application to Client Application

Purpose	Exercise the Card Application to perform Internal Authenticate.
Target	pivCrypt
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS03.04, AS04.01, AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application (required for step 1 for the '9A' PIV authentication key).
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<9A>> 3. Set algorithmIdentifier := <<06, 07 or 11>> 4. Set algorithmInput := <<Use the Dynamic Authentication Template format(Table 6 of SP 800-73-3 Part 2)to encode a challenge to be sent to the card>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle

	• *(IN)* keyReference • *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *(IN) inputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength* 7. If the PIV test card supports the 9E key, repeat step 1 – 6 but with the 9E key (Card Authentication key) and algorithmIdentifier := << '00', '03', '06, '07', '08', '0A', '0C' or '11'>> 8. If the PIV test card supports the 9C key, perform pivLogIntoCardApplication with correct PIN and repeat step 1 – 6, but with the 9C key (PIV Digital Signature Key), algorithm <<'07', '11' or '14'>> and data to sign instead of a challenge 9. If the PIV test card supports the 9D key, Repeat step 1 – 6, but with the 9D key (PIV Key Management Key), algorithm <<'07', '11' or '14'>> and an encrypted key (with algorithm '07') or a public key (with algorithms '11' or '14') instead of a challenge.
Expected Result(s)	Call returns with status_word of PIV_OK with the algorithmOutput carrying the encrypted challenge, transported key, shared secret Z, or a signature from the card.
Post Condition(s)	N/A

B.9.1.2 Mutual Authentication of Client Application and Card Application

Purpose	Exercise the Card Application to perform Mutual Authenticate.
Target	pivCrypt
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS03.04, AS04.01, AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<9B>> 3. Set algorithmIdentifier := <<00, 03, 08, 09, 0A,or 0C>> 4. Set algorithmInput := <<Use the Dynamic Authentication Template format (Table 6 of SP 800-73-3 Part 2) to first request a witness from the card, then followed by a second call that contains the decryption of the encrypted nonce from the card appended with the client's application generated nonce.>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* algorithmIdentifier

	• *(IN)* algorithmInput • *(IN) imputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength*
Expected Result(s)	1. The first Call returns with status_word of PIV_OK with the algorithmOutput carrying the encrypted nonce from the card. 2. The second call returns the status word of PIV_OK with algorithmOutput carrying the encrypted text of the client application generated nonce.
Post Condition(s)	Client Application set (updated) application security status with the selected PIV Card Application.

B.9.1.3 Authenticate the Client Application to Card Application

Purpose	Exercise the Card Application to perform External Authenticate.
Target	pivCrypt
Reference(s)	1. SP 800-73 Part 3, Section 3.3.1 2. AS03.04, AS04.01, AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<9B>> 3. Set algorithmIdentifier := <<00, 03, 08, 0A, or 0C 4. Set algorithmInput := <<Use the Dynamic Authentication Template format(Table 6 of SP 800-73 Part 2)to first request a challenge and then to encode an encrypted challenge in the next call>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *(OUT)* algorithmOutput • *(INOUT) outputLength*
Expected Result(s)	1. The first Call returns with status_word of PIV_OK with the algorithmOutput carrying the challenge from the card. 2. The second call returns the status word of PIV_OK.
Post Condition(s)	The Client Application set (updated) application security status with the selected PIV Card Application.

B.9.2 Test Assertions for Error Conditions

B.9.2.1 Identify and handle invalid card handles

Purpose	Ensure the Client Application can detect invalid card handles.
Target	pivCrypt
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application.
Test Steps	1. Set cardHandle := <<an invalid cardHandle>> 2. Set keyReference := <<a recognized key reference>> 3. Set algorithmIdentifier := <<a recognized Algorithm Identifier>> 4. Set algorithmInput := <<byte sequence compatible with the chosen algorithm identifier AND keyReference>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *(IN) imputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_CARD_HANDLE
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivCrypt function call. 2. The precondition states are unaffected.

B.9.2.2 Identify and handle invalid key reference or algorithm

Purpose	Ensure the Client Application can detect invalid key reference or algorithm.
Target	pivCrypt
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application.
Test Steps	1. Set cardHandle := <<a valid cardHandle>> 2. Either the keyReference (or) algorithmIdentifier (or) both set to an invalid value. 3. Set algorithmInput := <<byte sequence compatible with the type of authentication encoded according to the format in the Dynamic Authentication Template – Table 6 of SP 800-73-3 Part 2>> 4. Create algorithmOutput reference 5. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference

	• *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *(IN) imputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength*
Expected Result(s)	Call returns with status_word of PIV_INVALID_KEYREF_OR_ALGORITHM
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivCrypt function call. 2. The precondition states are unaffected.

B.9.2.3 Identify and handle invalid input data

Purpose	Ensure that the Client Application can identify and handle input data (algorithmInput) that is not compatible with the requested algorithm/key combination.
Target	pivCrypt
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is logged into the Card Application.
Test Steps	1. Set cardHandle := <<a valid cardHandle>> 2. Set keyReference := <<a key reference compatible with the algorithmIdentifier input value>> 3. Set algorithmIdentifier := <<a recognized Algorithm Identifier>> 4. Set algorithmInput := << byte sequence not compatible with the type of authentication and not encoded according to the format in the Dynamic Authentication Template – Table 6 of SP 800-73-3 Part 2>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *((IN) imputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength*
Expected Result(s)	Call returns with status_word of PIV_INPUT_BYTES_MALFORMED
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivCrypt method call. 2. The precondition states are unaffected.

B.9.2.4 Security Conditions are Enforced

Purpose	Verify that Internal Authenticate is performed with enforced security

	conditions (with/without logging (PIN VERIFY)) into the Card Application (see Table 3, Part 1 and Table 2, Part 2 (General Authenticate) security condition requirements)
Target	`pivCrypt`
Reference(s)	1. SP 800-73-3 Part 3, Section 3.3.1 2. AS04.08
Precondition(s)	1. The Client Application currently owns a connection to the Card Application accessible through cardHandle. 2. The client is not logged into the Card Application.
Test Steps	1. Set cardHandle := <<valid cardHandle>> 2. Set keyReference := <<9A>> 3. Set algorithmIdentifier := <<06, 07 or 11>> 4. Set algorithmInput : = <<Use the Dynamic Authentication Template format (Table 6 of SP 800-73-3 Part 2) to encode a challenge to be sent to the card>> 5. Create algorithmOutput reference 6. Call pivCrypt w/ • *(IN)* cardHandle • *(IN)* keyReference • *(IN)* algorithmIdentifier • *(IN)* algorithmInput • *(IN) inputLength* • *(OUT)* algorithmOutput • *(INOUT) outputLength* 7. If the PIV test card supports the 9E key, repeat step 1 – 6 but with the '9E' key reference (Card Authentication key) and algorithmIdentifier := << '00', '03', '06','07', 08', '0A', '0C'or '11' 8. If the PIV test card supports the 9C key, repeat step 1 – 6, but with the '9C' key reference (PIV Digital Signature Key) and algorithmIdentifier <<'07', '11' or '14'>> 9. If the PIV test card supports the 9D key, Repeat step 1 – 6, but with the 9D key reference (PIV Key Management Key) and algorithmIdentifier << '07', '11' or '14'>>
Expected Result(s)	• Step 7 Call returns with status_word of `PIV_OK` with the algorithmOutput carrying the encrypted challenge from the card • All other calls return with status_word of `PIV_SECURITY_CONDITIONS_NOT_SATISFIED`
Post Condition(s)	1. The Card Application returns to the state it had prior to the pivCrypt method call. 2. The precondition states are unaffected.

B.10 pivMiddlewareVersion

B.10.1 Valid Test Assertions

B.10.1.1 Retrieve the supported PIV MiddlwareVersion

Purpose	Ensure the Client Application can retrieve the PIV Middleware version from the PIV Middleware.
Target	`pivMiddlewareVersion`
Reference(s)	SP 800-73-3 Part 3, Section 3.1.1 AS03.04, AS04.01, AS04. 11
Precondition(s)	N/A.
Test Steps	• `Call pivMiddlewareVersion w/` • *`(OUT) version`*
Expected Result(s)	Function call returns with the version string "80073-3 Client API"
Post Condition(s)	N/A

Appendix C—Card Command Interface Test Assertions

Test Assertion Template

Purpose	A quick description of the test and why it is being run
Reference(s)	1. References to the SP 800-73-3 or other relevant publications 2. References to DTRs
Precondition(s)	Anything that must be done or known prior to executing the scenario
Test Scenario	`Sequence of APDU calls`
Expected Result(s)	What the expected execution path yields in terms of progress and values
Post Condition(s)	A description of the card application state once the test scenario completes

C.1 Card Commands for Data Access

C.1.1 SELECT Card Command

C.1.1.1 Contact Interface

Purpose	Validates that the PIV Card executes the SELECT card command through the contact interface for the following conditions: 1. Long AID 2. Right-truncated short AID
Reference(s)	1. SP 800-73-3 Part 2, Section 3.1.1 2. AS01.04, AS01.05, AS01.06, AS01.07, AS01.08, AS03.02, AS05.01, AS05.05, AS05.06, AS05.07, AS05.08, AS05.09, AS05.10, AS05.11
Precondition(s)	1. A valid PIV Card is inserted into the contact reader 2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. No application is currently connected to the PIV Card Application

Test Scenario	`1. Send SELECT card command with,`
	• `AID == A0 00 00 03 08 00 00 10 00 01 00`
	
	`2. Send SELECT card command without the version number,`
	• `AID == A0 00 00 03 08 00 00 10 00`

Expected Result(s)	1. The command returns the Application Property Template (APT) with the status words “90 00” at the end. Check that the application property template conforms to Table 3 of SP 800-73-3 Part 2 2. The command returns the Application Property Template (APT) with the status words “90 00” at the end. Check that the

	application property template conforms to Table 3 of SP 800-73-3 Part 2
Post Condition(s)	PIV Card Application is now the currently selected application. The application security status of the PIV Card Application is established.

C.1.1.2 Error Condition

Purpose	Validates that the PIV Card Application is not deselected while the currently selected application is the PIV Card Application and the SELECT command is sent with an AID that is not supported by the card.
Reference(s)	1. SP 800-73-3 Part 2, Section 3.1.1 2. AS05.10
Precondition(s)	1. A valid PIV Card is inserted into the contact reader 2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. No application is currently connected to the PIV Card Application
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Repeat step 1 with, • AID == A0 00 00 03 08 00 00 00 00 (invalid AID) 3. Send GET DATA card command with, • Data field of the command containing the tag of the Card Capability Container data object (ALWAYS READ) (This test case is executed with both Long AID and Short AID)
Expected Result(s)	1. The command returns the Application Property Template (APT) with the status words "90 00" at the end 2. The command returns '6A 82', application not found 3. The command returns the Card Capability Container (CCC) object with the status words "90 00" at the end
Post Condition(s)	The PIV Card Application continues to be the currently selected application and the application security status of the PIV Card Application remains unchanged.

C.1.1.3 Contactless Interface

Purpose	Validates conformance of the SELECT card command through the contactless interface
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.01, AS05.05, AS05.07, AS05.08, AS05.10
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader

	2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send SELECT card command without the version number, • AID == A0 00 00 03 08 00 00 10 00 3. Repeat step 1 with, AID == A0 00 00 03 08 00 00 00 00 (invalid AID)
Expected Result(s)	1. The command returns the Application Property Template (APT) with the status words "90 00" at the end. The application property template conforms to Table 3 of SP 800-73-3 Part 2 2. The command returns the Application Property Template (APT) with the status words "90 00" at the end. The application property template conforms to Table 3 of SP 800-73-3 Part 2 3. The command returns '6A 82', application not found
Post Condition(s)	PIV Card Application is the currently selected application and the application security status of the PIV Card Application is established.

C.1.2 GET DATA card command

C.1.2.1 Contact Interface

Purpose	Validates that the PIV Card accepts the GET DATA command through the contact interface and with the access rule of each container as specified in Table 1 of SP 800-73-3 Part 1. This test is applicable to the mandatory data objects required by SP 800-73-3, and the optional data objects when supported by the card.
Reference(s)	1. SP 800-73-3 Part 2, Section 3.1.2 2. AS05.01, AS05.12, AS05.12A, AS.05.02
Precondition(s)	1. A valid PIV Card is inserted into the contact reader 2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. No application is currently connected to the PIV Card Application 4. The optional containers supported by the card are recorded
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send GET DATA command with, • Data field of the command containing the tag of the Card Capability Container data object

	3. Send GET DATA command with, • Data field of the command containing the tag of the CHUID data object 4. Send GET DATA command with, • Data field of the command containing the tag of the X.509 Certificate for PIV Authentication data object 5. Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Fingerprints data object 6. If the card application supports the Printed Information data object: Send GET DATA command with, • Data field of the command containing the tag of the Printed Information data object 7. If the card application supports the Cardholder Facial Image data object: Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Facial Image data object 8. If the card application supports the Cardholder Iris image data object: Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Iris image data object 9. If the card application supports the X.509 Certificate for Digital Signature data object: Send GET DATA command with, • Data field of the command containing the tag of the X.509 Certificate for Digital Signature data object 10. If the card application supports the X.509 Certificate for Key Management data object: Send GET DATA command with, • Data field of the command containing the tag of the X.509 Certificate for Key Management data object 11. If the card application supports the X.509 Certificate for Card Authentication data object: Send GET DATA command with, • Data field of the command containing the tag of the X.509 Certificate for Card Authentication data object 12. Send GET DATA command with,

	• Data field of the command containing the tag of the Security Object 13. If the card application supports the Discovery Object data object: Send GET DATA command with, • Data field of the command containing the tag of the Discovery Object data object 14. If the card application supports the Key History data object: A) Send GET DATA command with, • Data field of the command containing the tag of the Key History data object B) For each implemented (up to twenty) retired X.509 Certificates for Key Management, • Send GET DATA command with, Data field of the command containing the tag of a retired X.509 Certificate for Key Management data object 15. Send VERIFY card command with, • P2, key reference value is set to '80' • Data field of the command will contain the PIN value obtained from the vendor, and padded with 'FF' to complete the total length of the field to 8 bytes. (This command could additionally be executed with P2 = '00' if the card supports the Global PIN (as indicated by the values 0x6010 or 0x6020 in the PIN Usage Policy sub-element of the Discovery Object). The expected result is the same as when the key reference '80' is used. 16. Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Fingerprints data object 17. If the card application supports the Printed Information data object: Send GET DATA command with, • Data field of the command containing the tag of the Printed Information data object if supported by the card 18. If the card application supports the Cardholder Facial Image data object: Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Facial Image data object if supported by the card

	19. If the card application supports the Iris Image data object: Send GET DATA command with, • Data field of the command containing the tag of the Iris Image data object 20. Send GET DATA command with, • Data field of the command containing a tag that does not identify any of the data objects resident on the card
Expected Result(s)	1. The command returns the Application Property Template with the status words "90 00" at the end 2. The command returns the Card Capability Container data object along with the status words "90 00" at the end 3. The command returns the CHUID data object along with the status words "90 00" at the end 4. The command returns the X.509 Certificate for PIV Authentication along with the status words "90 00" at the end 5. For steps 5 through 8, the command returns "69 82", security status not satisfied due to lack of PIN entry 6. In step 9, the command returns the X.509 Certificate for Digital Signature with the status words "90 00" at the end 7. In step 10, the command returns the X.509 Certificate for Key Management with the status words "90 00" at the end 8. In step 11, the command returns the X.509 Certificate for Card Authentication with the status words "90 00" at the end 9. In step 12, the command returns the Security Object with the status words "90 00" at the end 10. In step 13, the command returns the Discovery Object with the status words "90 00" at the end 11. In step 14A, the command returns the Key History data object with the status words "90 00" at the end 12. In step 14B, the command returns the retired X.509 Certificates for Key Management data objects, each with status words "90 00" at the end 13. In step 15, the command returns the status words "90 00" 14. For steps 16 through 19 the command returns the requested data object along with the status words "90 00" at the end 15. In step 20, the command returns '6A 82', data object not found
Post Condition(s)	NA

C.1.2.2 Contactless Interface

Purpose	Validates the conformance of the GET DATA command through the

	contactless interface. This test is applicable to the mandatory data objects required by SP 800-73-3, and the optional data objects when supported by the card.
Reference(s)	1. Table 2 of SP 800-73-3 Part 2 2. AS05.01, AS05.02, AS05.12, AS05.12A
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader 2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader
Test Scenario	• `Repeat the steps 1-14 and step 20 from the Test C.1.2.1`
Expected Result(s)	1. The command returns the Application Property Template with the status words "90 00" at the end 2. In step 2, the command returns the status words "69 82", security status is not satisfied due to the contactless interface 3. In step 3, the command returns the CHUID data object along with the status words "90 00" at the end 4. For steps 4 through 10, the command returns "69 82", security status is not satisfied due to the contactless interface 5. In step 11, the command returns the X.509 Certificate for the Card Authentication Key Object with the status words "90 00" at the end 6. In step 12, the command returns the status words "69 82", security status is not satisfied due to the contactless interface 7. In step 13, the command returns the Discovery Object with the status words "90 00" at the end 8. For steps 14A and B, the command returns "69 82", security status is not satisfied due to the contactless interface 9. For referred step 20, the command returns "6A 82", Data Object not found
Post Condition(s)	NA

C.2 Commands for Authentication

C.2.1 VERIFY Card Command

C.2.1.1 Contact Interface

Purpose	Validates the following conditions associated with the VERIFY command: 1. Without an unsupported PIN 2. Successful execution of the command (with PIV Card Application PIN and (if supported) Global PIN) 3. Execution of the command with a PIN not formatted per SP

	800-73-3 4. Multiple execution of the command with an incorrect PIN (formatted correctly) until the retry counter reaches zero
Reference(s)	1. SP 800-73-3 Part 2, Section 3.2.1 2. AS01.17, AS05.01, and AS05.12 through AS05.22B
Precondition(s)	1. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application 2. Cardholder PIV Card Application PIN is recorded 3. Cardholder Global PIN (if supported) is recorded 4. PIV Application PIN and Global PIN (if implemented) reset retry counter value(s) (maximum number of PIN tries allowed) is recorded
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send GET DATA command with, • Data field of the command containing the tag of the Discovery Object • If the commands returns “6A 82”, “Data Object not Found” or the Discovery Object is returned and its PIN Usage Policy bytes is set to ’4000’, follow steps outlined in 2a • If the commands returns the Discovery Object and its PIN Usage Policy bytes is set to ’6010’ or ’6020’, follow steps in 2b, 2a. Test case for implementations that support only the PIV Card Application PIN for PIV data access and command execution. 1. Send VERIFY card command with, • P2, key reference value is set to a value other than what is supported by the PIV Card Application. • Data field of the command will contain a random PIN value. The PIN is either truncated or padded with ‘FF’ to complete the total length of the field to 8 bytes. 2. Send GET DATA command with, • Data field of the command containing the tag of the Cardholder Fingerprint data object 3. Send VERIFY card command with, • P2, key reference value is set to ‘80’ • Data field of the command will contain the correct PIV card application cardholder PIN value, and padded with ‘FF’ to complete the

total length of the field to 8 bytes.

4. Send GET DATA command with,
   - Data field of the command containing the tag of the Cardholder Fingerprint data object

5. Send VERIFY card command with,
   - P2, key reference value is set to '80'
   - Data field of the command will contain an arbitrary PIN value other than what is obtained from the vendor, NOT padded with 'FF' to complete the total length of the field to 8 bytes

6. Send VERIFY card command repeatedly, until after the issuer specified maximum number of PIN tries are exceeded with,
   - P2, key reference value is set to '80'
   - Data field of the command will contain an arbitrary PIN value other than what is obtained from the vendor, and padded with 'FF' to complete the total length of the field to 8 bytes

2b. Test case for implementations that support both the PIV Card Application PIN and the Global PIN (0x00) for PIV data access and command execution (as indicated by the Discovery Object's PIN Usage Policy values 60 10' or '6020').

1. Send VERIFY card command with,
   - P2, key reference value set to a value other than what is supported by the card
   - Data field of the command will contain the correct Global PIN value, and padded with 'FF' to complete the total length of the field to 8 bytes.

2. Send GET DATA command with,
   - Data field of the command containing the tag of the Cardholder Fingerprint data object

3. Send VERIFY card command with,
   - P2, key reference value is set to '00'
   - Data field of the command will contain the correct Global PIN value, and padded with 'FF' to complete the total length of the field to 8 bytes.

4. Send GET DATA command with,
   - Data field of the command containing the tag of the Cardholder Fingerprint data object

	5. Send VERIFY card command with, • P2, key reference value is set to '00' • Data field of the command will contain an arbitrary PIN value other than what is obtained from the vendor, NOT padded with 'FF' to complete the total length of the field to 8 bytes 6. Send VERIFY card command repeatedly, until after the issuer specified maximum number of PIN tries are exceeded with, • P2, key reference value is set to '00' • Data field of the command will contain an arbitrary PIN value other than what is obtained from the vendor, and padded with 'FF' to complete the total length of the field to 8 bytes 7. Reset the PIV Card Application 8. Perform steps 3 – 6 as outlined in 2a
Expected Result(s)	1. The command returns the Application Property Template with the status words "90 00" at the end 2: The command returns either 1) the Discovery Object with the status words "90 00" at the end or 2) '6A 82', data object not found 2a : 1. The command returns '6A 88' Key Reference not Found 2. The command returns "69 82' Security status not satisfied 3. The command returns '90 00' (verify that the retry counter is set to Reset Retry Value) 4. The command returns the cardholder fingerprint data object along with the status word '90 00' 5. The command returns '6A 80' (Incorrect parameter command data field) 6. The command returns • '63 CX' until the maximum number of PIN tries are reached (X indicates the number of further allowed retries) • '69 83' (Authentication method blocked) when the maximum number of PIN tries are exceeded 2b: Steps: 1 - 6 have the same command responses as in 2a (1 to 6) Steps 8 : Referred steps 3 to 6 have the same command responses as in 2a (steps 3 to 6)
Post Condition	The card is blocked

C.2.1.2 Contactless Interface

Purpose	Validates that the PIV Card does not accept the VERIFY command through the contactless interface
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.03, AS05.04
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader 2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader 4. The PIV Card Application is the currently selected application
Test Scenario	1. Send VERIFY card command with, • P2, key reference value is set to '80' • Data field of the command will contain the correct cardholder PIN value, and padded with 'FF' to complete the total length of the field to 8 bytes 2. Perform step 2 in C.2.1.1 If the result of step 2 branches to step 2b: 3. Send VERIFY card command with, • P2, key reference value is set to '00' • Data field of the command will contain the correct cardholder Global PIN value, and padded with 'FF' to complete the total length of the field to 8 bytes
Expected Result(s)	1. The command in step 1 returns '6A 81' (Function not supported) 2. The command in step 3 returns '6A 81' (Function not supported)
Post Condition(s)	NA

C.2.2 CHANGE REFERENCE DATA card command

C.2.2.1 Contact Interface

Purpose	Validates that the PIV Card executes the CHANGE REFERENCE DATA command for the following conditions: 1. Without the proper security condition (PIV Card Application PIN and (if supported) Global PIN) 2. After the security condition is satisfied 3. With an incorrect PIN until the retry counter reaches zero
Reference(s)	1. SP 800-73-3 Part 2, Section 3.2.2, 2. AS05.01, AS05.23 through AS05.28A,
Precondition(s)	1. PIV Application PIN and Global PIN (if supported) reset retry counter value (maximum number of PIN tries allowed) is recorded 2. Cardholder PIV Card Application PIN is recorded 3. Cardholder Global PIN (if supported) is recorded

	4. A valid PIV Card is inserted into the contact reader 5. There exists a valid PC/SC connection between the test system and an instance of the contact reader 6. No application is currently connected to the PIV Card Application
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send CHANGE REFERENCE DATA card command with, • P2, key reference value, is set to a value other than what is supported by the card • Data field of the command will contain the correct PIN value (PIN 2) concatenated without delimitation with an arbitrary new PIN value (PIN 3). Both PINs should be truncated or padded with 'FF' to complete the total length of the field to 8 bytes 3. Perform step 2 in C.2.1.1 (This step reads the Discovery Object from the card and parses the PIN usage Policy sub-element) • If the result from step 2 in C.2.1.1 branches to step 2a, perform the steps in 3a (This is the case when the discovery object's PIN usage policy indicates no support for the Global PIN (0x40 00) • If the result from step 2 of C.2.1.1 branches to step 2b, but the implementation does not support CHANGE REFERENCE DATA APDU with the Global PIN, perform the steps in 3a • (This is the case when the discovery object's PIN usage policy indicates support for the Global PIN (0x6010 or 0x6020), but vendor documentation does not support the use of CHANGE REFERENCE DATA command with the Global PIN) • If the result from step 2 of C.2.1.1 branches to 2b and CHANGE REFERENCE DATA command with the Global PIN (0x00) is implemented with the PIV Card Application, perform steps 3b (This is the case when the discovery object's PIN usage policy indicates support for the Global PIN (0x6010 or 0x6020), and vendor documentation indicates the support of CHANGE REFERENCE DATA command with the Global PIN) 3a: Test case for implementations supporting only the PIV Card Application PIN to be changed through the CHANGE REFERENCE DATA command

	1) Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '80' • Data field of the command will contain the correct PIN value (PIN 1) obtained from the vendor, concatenated without delimitation with an arbitrary new PIN value (PIN 2). Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes 2) Send VERIFY card command with, • P2, key reference value is set to '80' • Data field of the command will contain the new PIN value (PIN 2 from previous step), and padded with 'FF' to complete the total length of the field to 8 bytes. 3) Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '80' • Data field of the command will contain the correct PIN value (PIN 2) concatenated without delimitation with an arbitrary new PIN value (PIN 3) that is less than 8 bytes 4) Send CHANGE REFERENCE DATA card command repeatedly, until after the issuer specified maximum number of PIN tries are exceeded with, • P2, key reference value is set to '80' • Data field of the command will contain an incorrect PIN value (anything other than PIN 2), concatenated without delimitation with an arbitrary new PIN value (PIN 3)until after the issuer specified maximum number of tries are exceeded. Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes. 3b: Test case for implementations that support the PIV Card Application PIN and Global PIN (0x00)to be changed through the CHANGE REFERENCE DATA command. 1) Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '00' • Data field of the command will contain the correct PIN value (PIN 1) obtained from the vendor, concatenated without delimitation with an arbitrary new PIN value (PIN 2). Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes 2) Send VERIFY command with Send VERIFY card command with, • P2, key reference value is set to '00' • Data field of the command will contain the new

	PIN value (PIN 2 from previous step), and padded with 'FF' to complete the total length of the field to 8 bytes. 3) Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '00' • Data field of the command will contain the correct PIN value (PIN 2) concatenated without delimitation with an arbitrary new PIN value (PIN 3) that is less than 8 bytes 4) Send CHANGE REFERENCE DATA card command repeatedly, until after the issuer specified maximum number of PIN tries are exceeded with, • P2, key reference value is set to '80' • Data field of the command will contain an incorrect PIN value (anything other than PIN 2), concatenated without delimitation with an arbitrary new PIN value (PIN 3) until after the issuer specified maximum number of tries are exceeded. Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes. 5) Perform steps 1 to 4 of 3a.
Expected Result(s)	1. Command returns the APT with the status words '90 00' at the end 2. Command returns '6A 88' (Reference Data not found) 3a: 1) The Command returns '90 00' (Also Verify that the retry counter is set to Reset Retry Value) 2) The Command returns '90 00' 3) The command returns '6A 80' (Incorrect parameter in command data field) 4) The command returns: • '63 CX' until the maximum number of tries are reached. (X indicates the number of further allowed retries) • '69 83' (Authentication method blocked) when the maximum number of tries are exceeded 3b: Steps 1 – 4 have the same command responses as in 3a, steps 1 – 4 Steps 5: Referred steps 1 - 4 have the same command responses as in 3a steps 1 - 4
Post Condition(s)	The card is blocked.

C.2.2.2 Contactless Interface

Purpose	Validates that the PIV Card does not accept the CHANGE REFERENCE DATA command through the contactless interface
Reference(s)	SP 800-73-3 Part 2, Table 2 1. AS05.03
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader 2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '80' • Data field of the command will contain the correct PIN value (PIN 1) obtained from the vendor, concatenated without delimitation with an arbitrary new PIN value (PIN 2). Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes Perform step 2 in C.2.1.1 • If the result of step 2 branches to step 2b and the implementation support CHANGE REFERENCE DATA APDU with the Global PIN: 3.Send CHANGE REFERENCE DATA card command with, • P2, key reference value is set to '00' • Data field of the command will contain the correct PIN value (PIN 1) obtained from the vendor, concatenated without delimitation with an arbitrary new PIN value (PIN 2). Both PINs should be padded with 'FF' to complete the total length of the field to 8 bytes
Expected Result(s)	1. Command returns the APT with the status words '90 00' at the end 2. Command returns the status words '6A 81' (Function not supported) 3. Command returns the status words '6A 81' (Function not supported)
Post Condition(s)	PIN remains unchanged

C.2.3 RESET RETRY COUNTER command

C.2.3.1 Contact Interface

Purpose	Validates that the PIV Card executes the RESET RETRY COUNTER command for the following conditions: 1. Without the security condition satisfied 2. After the security condition (authenticated with the PUK) is satisfied 3. With a valid new PIN value not formatted per SP 800-73-3 4. With a valid new PIN value (formatted correctly) 5. With a valid new PIN value causing the PUK retry counter to be optionally reset 6. Without the security condition satisfied (incorrect PUK) until RESET RETRY COUNTER command is blocked
Reference(s)	1. SP 800-73-3 Part 2, Section 3.2.3 2. AS05.01, AS03.07, AS05.29 through AS05.33
Precondition(s)	1. PIV Application PIN reset retry counter value (maximum number of PIN tries allowed) is recorded 2. There exists a valid physical connection between an instance of the PIV Card and the host of the calling application and PIV Card Application is the currently selected application. 3. The initial value of the retry counter associated with the PIV Card application PIN is as stated by the vendor/issuer 4. The value of the counter reference data (PUK) is as stated by the vendor/issuer
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Send RESET RETRY COUNTER with, • P2, key reference value, is set to a value other than what is supported by the card • Data field of the command contains the PUK value for key reference '80', concatenated without delimitation with a valid new PIN and each padded with 'FF' to complete the total length of the field to 8 bytes 3. Repeat Step 2 with, • P2, key reference value is set to '80' • Data field of the command contains the PUK value for key reference '80' concatenated with the new PIN value that is not padded to complete 8 bytes 4. Repeat step 2 with: • P2, key reference value is set to '80' • Data field of the command contains the PUK value for key reference '80' concatenated without delimitation with the new PIN and each padded with 'FF' to complete

	the total length of the field to 8 bytes Perform steps 5 - 7, only if the reset of the PIN's retry counter also resets the PUK retry counter: 5.Repeat step 2 with • P2, key reference value is set to '80' • Data field of the command containing an incorrect PUK value for key reference '80' concatenated without delimitation with a new valid PIN value and each padded with 'FF' to complete the total length of the field to 8 bytes. Record the number of remaining retries X in return code 63 CX. 6. Repeat step 4 7. Repeat step 2 with • P2 key reference value is set to '80' • Data field of the command containing an incorrect PUK value for key reference '80' concatenated without delimitation with a new valid PIN and each padded with 'FF' to complete the total length of the field to 8 bytes. Record the number of remaining retries X in return code 63 CX. 8. Repeat step 2 with • P2, key reference value is set to '80' • Data field of the command containing an incorrect PUK value concatenated without delimitation with the new PIN and each padded with 'FF' to complete the total length of the field to 8 bytes. This operation is repeated until the number of resets allowed is exceeded.
Expected Result(s)	1. Command returns the APT with the status words '90 00' at the end 2. The command returns '6A 88' (key reference not found) 3. The command returns ' '6A 80' (incorrect parameter in command data field) 4. The command returns '90 00'. Validate that the existing PIN is changed to new PIN (say PIN 2). Also Verify that the PIN's retry counter is set to Reset Retry Value 5. The command returns '63CX' (X == number of reset left) 6. The command returns '90 00' 7. The command returns '63CX. Verify that X from this step >= X from step 5 8. The command returns • '63 CX' , (X==number of resets left) • '69 83' (Authentication method blocked) – when the command is invoked after the value of X becomes zero.

	NOTE: Testing this condition may leave the card unusable in some implementations for all operations related to the key reference associated with this reset counter.
Post Condition(s)	1. No further resets of reference data associated with key reference possible.

C.2.3.2 Contactless Interface

Purpose	Validates that the RESET RETRY COUNTER command cannot be issued through the contactless interface
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.03
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader 2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Repeat steps 1 and 4 of test C.2.3.1
Expected Result(s)	1. Step 1 referenced above returns the APT with the status words '90 00' at the end 2. Step 4 referenced above returns '6A 81' (Function not supported)
Post Condition(s)	1. Reference data associated with key reference is not changed. Retry counter value associated with the key reference is not reset. The Reset counter value is unchanged.

C.2.4 GENERAL AUTHENTICATE card command

C.2.4.1 Contact Interface

Purpose	Validates the GENERAL AUTHENTICATE command to : 1. Authenticate the PIV Card Application to the Test Toolkit Application (INTERNAL AUTHENTICATE) 2. Authenticate the Client Application (External Authentication) 3. Two way authentication of PIV Card Application and Test Toolkit Application (MUTUAL AUTHENTICATE) 4. Sign with the 9C PIV Digital Signature private key 5. Enable Key Establishment functionality with the 9D PIV Key Management private key 6. Enable Key History Mechanism functionality with retired private Key Management Keys
Reference(s)	1. SP 800-73-3 Part 2, Section 3.2.4 2. AS05.01, AS03.06, AS05.25, AS034 through AS05.36
Precondition(s)	1. A valid PIV Card is inserted into the contact reader

	2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. The PIV Card Application is the currently selected application on the card 4. The length of the challenge supported by the card is obtained 5. The PIN security condition is satisfied (required for private key reference value 9A, 9C and 9D)
Test Scenario	1. (Internal Authenticate using an asymmetric key) Send GENERAL AUTHENTICATE card command • CLA is set to: • '00' if command chaining is not needed or • '10' if command chaining is used. (The last chain of the command sets CLA to '00') • P1, algorithm reference, is set to '06', '07' or '11' • P2, key reference, is set to '9A' indicating the PIV Authentication Key • Data field in the command is to include '81' specifying a challenge, followed by a randomly generated challenge NOTE: The following test invocation (step 2) is to be performed only if the PIV Card Application supports the Card Authentication key and the key is a symmetric key. 2. (Internal Authenticate using a symmetric key) Send GENERAL AUTHENTICATE card command • CLA is set to: • '00' if command chaining is not needed or • '10' if command chaining is used. (The last chain of the command sets CLA to '00') • P1, algorithm reference, is set to '00', '03','08', '0A' or '0C', • P2, key reference, is set to '9E' indicating the PIV Card Authentication Key • Data field in the command is to include '81' specifying a challenge, followed by a randomly generated challenge NOTE: The following two test invocations (3a and 3b) are to be performed only if the PIV Card Application supports the use of the key '9B'. 3a. (Mutual Authenticate using a symmetric key) Send GENERAL AUTHENTICATE card command • CLA is set to: • '00' if command chaining is not needed or • '10' if command chaining is used. (The last chain of the command sets CLA to '00') • P1, algorithm reference, is set to '00', '03', '08', '0A' or '0C', • P2, key reference, is set to '9B'

- Data field in the command is to include ‘80’ requesting a witness from the PIV Card application.

3b. (Mutual Authenticate using a Symmetric key) Send GENERAL AUTHENTICATE card command

- P1, algorithm reference is set to the same value as specified in the previous step (3a)
- P2, key reference is set to ‘9B’
- Data field in the command is to include ‘80’ followed by decryption of the encrypted nonce sent by the card application and ‘81’ followed by another nonce.

NOTE: The following two test invocations are to be performed only if the PIV Card Application supports the use of the key ‘9B’.

4. (External Authenticate using a symmetric key)
4a. Send GENERAL AUTHENTICATE card command

- CLA is set to 00
- P1, algorithm reference, is set to ‘00’, ‘03’, ’08’, ‘0A’ or ‘0C’
- P2, key reference, is set to ‘9B’ indicating the PIV Card Authentication Key
- Data field in the command is to include ‘81’ followed by 00 indicating it is a request for challenge.

4b. (Continue ..External Authenticate using a symmetric key) Send GENERAL AUTHENTICATE card command

- CLA is set to 00
- P1, algorithm reference, is set to the same value as in step 4a.
- P2, key reference, is set to same value as in 4a
- Data field in the command is to include ‘82’ followed by encrypted challenge.

NOTE: The following two test invocations (3a and 3b) are to be performed only if the PIV Card Application supports the use of the key ‘9E’.

5. Repeat step 3 a) and b) with P2 set to ‘9E’ (Card Authentication Key) and P1 (algorithm reference) set to ’06’, ‘07’or ‘11’

6a. If the ‘9C’ key is supported, perform step 3) of 2a in C.2.1.1 to verify cardholder’s PIN and repeat step 1 with P2 set to ’9C’, P1 (algorithm reference) set to ‘07’, ‘11’, or ‘14’ and template ‘81’ in the data field containing a hashed message.

6b. If the ‘9C’ key is supported, repeat step 1 (without PIN verification. Set P2 to ‘9C’, P1 (algorithm reference) to ‘07’, ‘11’, or ‘14’ and include template ‘81’ in the data

field containing a hashed message.

7. If the '9D' key is supported, perform step 3) of 2a in C.2.1.1 to verify cardholder's PIN and repeat step 1 with P2 set to '9D', P1 (algorithm reference) set to '07', '11' or '14' and include template '81' containing an encrypted key (in case of P1 ='07') or template '85'containing the other party's public key[6] (in case of P1 = '11' or '14').

8. If the 9E key is supported and is an asymmetric key, repeat step 1 with P2 set to '9E' (Card Authentication Key) and P1 (algorithm reference) set to '06', '07'or '11' and the template '81' containing a randomly generated challenge

9. If the Key History Data Object is supported:

Send GET DATA command with,

- Data field of the command containing the tag of the Key History data object. Retrieve the Key History's data elements:

- If keysWithOnCardCerts = 0 and keysWithOffCardCerts > 0

    - Read the certificate(s) and key references (pairs) from the vendor provided URL file. For each key reference value in the range (0x95 – keysWithOffCardCerts + 1) through 0x95, verify that the provided URL file includes that key reference, issue a challenge for that key reference, and verify the response using the public key from the corresponding certificate from the provided URL file.

- If keysWithOnCardCerts > 0 and keyWithOffCardCerts = 0

    - For each key reference value in the range 0x82 through (0x82 + keysWithOnCardCerts – 1), read the certificates from the card. Issue a challenge and responses for each retired private key[7].

- If keysWithOnCardCerts > 0 and keyWithOffCardCerts > 0

    - For each key reference value in the range 0x82 through (0x82 + keysWithOnCardCerts – 1) and in

[6] Template '85' contains the other party's public key (point) encoded as '04' || X || Y , where '04 || X || Y' is the other party's public key, a point on Curve P-256 or P-384, encoded without the use of point compression as described in Section 2.3.3 of [7].

[7] See table 6 of SP 800-73-3 for the association of each Certificate BER TLV tags to corresponding key reference values

	the range (0x95 – keysWithOffCardCerts + 1) through 0x95, verify that the provided URL file includes that key reference, issue a challenge for that key reference, and verify the response using the public key from the corresponding certificate from the provided URL file. 10. Repeat Step 1 with an invalid value of algorithm reference (P1) and/or key reference (P2). 11. Repeat Step 1 with an invalid value in Data field (improper challenge length for the chosen algorithm) 12. The PIN Security condition in the Pre-condition is annulled (by performing VERIFY with a wrong PIN) and 13. Repeat step 1 14. If the key types are supported in steps 2, 3, 4, 5, 7 8 and 9 repeat these steps
Expected Result(s)	1. The command returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card 2. The command returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card 3a. The PIV Card Application returns with the encryption of a nonce followed by '90 00' 3b. The PIV Card Application verifies the witness and then responds with encryption of the nonce sent by test toolkit application followed by '90 00' 4a. The PIV Card Application returns a nonce followed by '90 00' 4b. The Test Toolkit application responds with encryption of the nonce sent by PIV Card application and the card returns '90 00' 5. For step 3a, the PIV Card Application returns with the encryption of a nonce followed by '90 00' For step 3b. The PIV Card Application verifies the witness and then responds with encryption of the nonce sent by test toolkit application followed by '90 00' 6a: The command returns the signed data with '90 00' at the end. Verify the signature by applying the public key to the signed data and compare the result to the hash sent to the card 6b: The command returns '69 82' (Security status not satisfied) 7. For algorithm reference '07' as P1 value, the command returns the transported key with '90 00' at the end. Compare the plaintext key to the one received in the response from the card. For algorithm

	reference '11' or '14' as P1 value, the command returns the shared secret Z [8] with '90 00' at the end. Compare the shared secret computed by the card with the shared secret computed off-card 8. The command returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card 9. The command returns either 1) the transported key with '90 00' at the end or 2) the shared secret Z with '90 00' at the end. • For Key Transport (as indicated by algorithm reference '07' as P1 value), the command returns the transported key with '90 00' at the end. Compare the plaintext key to the one received in the response from the card. • For EC DH, (as indicated by algorithm reference '11' or '14' as P1 value), the command returns the shared secret Z [9] with '90 00' at the end. Compare the shared secret computed by the card with the shared secret computed off-card 10. The command returns '6A 86' (Incorrect parameter in P1 or P2) 11. The command returns '6A 80' (Incorrect parameter in command data field) 12. The security state is reset 13. The command returns '69 82' (Security status not satisfied) 14. The command returns: • For referenced step 2, the command returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card • '69 82' (Security status not satisfied) for referenced steps, 3, 5, 7 and 9 • Referenced step 4a: The PIV Card Application returns a nonce followed by '90 00' • Referenced step 4b. The Test Toolkit application responds with encryption of the nonce sent by PIV Card application and the card returns '90 00' • Referenced step 8: The command returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card

[8] Z is the X coordinate of point P as defined in SP 800-56A, Section 5.7.1.2
[9] Z is the X coordinate of point P as defined in SP 800-56A, Section 5.7.1.2

	Note on Steps 1, 6a, 8 and 14(8): If ECDSA with algorithm '11' (in case of 9C or 9E) or '14 ' (in case of 9C) is used, the response data field contains r and s[10]
Post Condition(s)	N/A

C.2.4.2 Contactless Interface

Purpose	Validates internal authentication and mutual authentication of the PIV card and the Test Toolkit follow the Contact only /Contactless interface mode for the private key in use.
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.03
Precondition(s)	1. A valid PIV Card is placed within the reading range of the contactless reader 2. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 3. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Repeat step 1 of C.2.4.1 3. If the key types are supported in C.2.4.1 (steps 2, 3, 4, 5, 6, 7, 8 and 9), repeat these steps
Expected Result(s)	1. The Command returns the APT with the status words '90 00' at the end 2. The command returns '6A 81' – Function not supported 3. Referenced Steps: • # 2 referenced in C.2.4.1 returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted challenge and compare it to the one sent to the card • # 3 referenced in C.2.4.1 returns '69 82' – Security Condition not satisfied • # 4a and 4b have the same response as 4a and 4b in C.2.4.1 • # 5, 6 and 7 returns '6A 81' – Function not supported • # 8 referenced in C.2.4.1 returns the encrypted challenge with '90 00' at the end. Decrypt the encrypted

[10] r and s are DER encoded with the following ASN.1 structure:

```
Ecdsa-Sig-Value ::= SEQUENCE {
        r     INTEGER,
        s     INTEGER  }
```

	challenge and compare it to the one sent to the card • # 9 referenced in C.2.4.1 returns '69 82' – Security Condition not satisfied Note for steps 8 and 13 (8): If ECDSA with algorithm '11' is used, the response data field contains the uncompressed form of r and s.
Post Condition(s)	N/A

C.3 Card Commands for Credential Initialization and Administration

C.3.1 PUT DATA Command

C.3.1.1 Contact Interface

Purpose	Validates that the PUT DATA command exhibits the appropriate behavior under the following conditions: 1. Without the security condition is satisfied 2. After the security condition is satisfied
Reference(s)	1. SP 800-73-3, Part 2, Section 3.3.1 2. AS05.01, AS05.02, AS05.37
Precondition(s)	1. A valid PIV Card is inserted into the contact reader 2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. The PIV Card Application is the currently selected application on the card 4. The mutual authentication of PIV Card Application and the test toolkit application has not been performed
Test Scenario	NOTE: The following tests (1 through 13) are run based on the assumption that the PIV Card Application either does not support the key referenced by '9B' or mutual authentication of the PIV Card Application and the test toolkit application has not been performed using steps 3a and 3b of C.2.4.1 (GENERAL AUTHENTICATE) 1. Send PUT DATA card command with, • CLA is set to: • '00' if command chaining is not needed or • '10' if command chaining is used. (The last chain of the command sets CLA to '00') • Data field in the command is to include the tag of the Card Capability Container object • Data field in the command is to include the data

	that will replace the CCC 2. Repeat step 1 with • Data field in the command is to include the tag of the CHUID object • Data field in the command is to include the data content that will replace the CHUID 3. Repeat step 1 with • Data field in the command is to include the tag of the PIV Authentication Certificate object • Data field in the command is to include data content that will replace the PIV Authentication Certificate 4. Repeat step 1 with • Data field in the command is to include the tag of the Cardholder Fingerprints data object • Data field in the command is to include data content that will replace the Cardholder Fingerprints 5. If the card supports the Printed Information, repeat step 1 with • Data field in the command is to include the tag of the Printed Information object • Data field in the command is to include the data content that will replace the Printed Information 6. If the card supports the Cardholder Facial Image, repeat step 1 with • Data field in the command is to include the tag of the Cardholder Facial Image object • Data field in the command is to include the data content that will replace the Cardholder Facial Image 7. If the card supports the certificate for Digital Signature, repeat step 1 with • Data field in the command is to include the tag of the certificate for Digital Signature object • Data field in the command is to include the data content that will replace the certificate for Digital Signature 8. If the card supports the X.509 Certificate for Key Management data object, repeat step 1 with • Data field in the command is to include the tag of the X.509 Certificate for Key Management data object • Data field in the command is to include the data content that will replace the X.509 Certificate for Key Management data object 9. If the card supports the X.509 Certificate for Card Authentication data object, repeat step 1 with • Data field in the command is to include the tag of

	the X.509 Certificate for Card Authentication data object • Data field in the command is to include the data content that will replace the X.509 Certificate for Card Authentication data object 10. If the card supports the Discovery Object, repeat step 1 with • CLA = 00 • Data field in the command is to include the tag of the Discovery object • Data field in the command is to include the data content that will replace the Discovery Object 11. Repeat step 1 with • Data field in the command is to include the tag of the Security Object • Data field in the command is to include the data content that will replace the Security Object 11. If the card supports the Key History Object, repeat step 1 with • Data field in the command is to include the tag of the Key History object • Data field in the command is to include the data content that will replace the Key History Object 12. If the card supports Key History Data Object, repeat step 1 for each implemented retired Key Management Key's X.509 Certificate with • Data field in the command is to include the tag of one of the 20 retired X.509 Certificates for Key Management • Data field in the command is to include the data content that will replace the retired X.509 Certificate for Key Management 13. If the card supports the Iris Image data object, repeat step 1 with • Data field in the command is to include the tag of the Iris Image data object • Data field in the command is to include the data content that will replace the Iris Image data object NOTE: The following tests are to be performed only if the PIV Card Application supports the use of the key '9B' 14. Perform mutual authentication of PIV Card Application and the test toolkit application using steps 3a and 3b of C.2.4.1 (GENERAL AUTHENTICATE) 15. Repeat steps 1-13 with GET DATA command immediately following each PUT DATA and verifying whether the same data that is input is returned.

Expected Result(s)	1. In Steps 1 through 13, commands return '69 82', (security status not satisfied) and the contents of the data objects remained unchanged 2. The two test invocations referred to in Step 14 should return the same responses as 3a and 3b of Expected Results under test C.2.4.1 3. In step 15, all commands return '90 00', and input and output data strings match.
Post Condition(s)	1. The contents of each object have been overwritten with the new values provided in step 15

C.3.1.2 Contactless Interface

Purpose	Validates that the PUT DATA command cannot be issued through the contactless interface
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.03
Precondition(s)	1. Record the existing values of all data objects 2. A valid PIV Card is placed within the reading range of the contactless reader 3. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 4. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Repeat step 2 of 3.1.1
Expected Result(s)	1. The command returns the APT 2. The command return '6A 81' – Function not supported
Post Condition(s)	1. The data container values remain unchanged

C.3.2 GENERATE ASYMMETRIC KEY PAIR command

C.3.2.1 Contact Interface

Purpose	Validates that the card executes the GENERATE ASYMMETRIC KEY PAIR command for the following conditions: 1. Without the security condition satisfied 2. After the security condition (authenticating with the PIV Card Application Administrator) is satisfied
Reference(s)	1. SP 800-73-3, Section Part 2, 3.3.2 2. AS05.01, AS05.38 through AS05.40
Precondition(s)	1. A valid PIV Card is inserted into the contact reader

	2. There exists a valid PC/SC connection between the test system and an instance of the contact reader 3. The PIV Card Application is the currently selected application on the card 4. The length of the challenge supported by the card is obtained
Test Scenario	NOTE: The following test (# 1) is run based on the assumption that the PIV Card Application either does not support the key '9B' or mutual authentication of the PIV Card Application and the test toolkit application has not been performed using steps 3a and 3b of C.2.4.1 (GENERAL AUTHENTICATE) 1. Send GENERATE ASYMMETRIC KEY PAIR card command with, • P2 is set to value '9A' • Data field in the command is to include either '06', '07' or '11' as the Cryptographic Mechanism Identifier NOTE: The following tests are to be performed only if the PIV Card Application supports the use of the key '9B'. 2. Perform mutual authentication of PIV Card Application and the test toolkit application using steps 3a and 3b of C.2.4.1 (GENERAL AUTHENTICATE) 3. Repeat Step 1 4. If the '9E' key is supported and is an asymmetric key (pair), repeat Step 1 with • P2 set to '9E' • Data field in the command is to include either '06', '07' or '11' as the Cryptographic Mechanism Identifier 5. If The '9C' key is supported, repeat Step 1 with • P2 set to '9C' • Data field in the command is to include either '07', '11', '14' as the Cryptographic Mechanism Identifier 6. If The '9D' key is supported and generated on card, repeat Step 1 with • P2 set to '9D' • Data field in the command is to include '07', '11; or '14' as the Cryptographic Mechanism Identifier. 7. Repeat Step 1 with the "Cryptographic Mechanism Identifier" value in the data field is set to a value that is not supported by the card. 8. Repeat Step 1 with P2 set to a key reference value that is not supported by the card or does not match up with the cryptographic algorithm specified in the data field.

Expected Result(s)	1. Command returns '69 82' (Security status not satisfied) 2. The two test invocations referred to in Step 2 should return the same responses as 3a and 3b of Expected Results under test C.2.4.1 3. Command returns the data object consisting of the '7F49' template with the generated public key and modulus (RSA), or point (ECDSA) followed by '90 00'. 4. Command returns the data object consisting of the '7F49' template with the generated public key and modulus (RSA), or point (ECDSA) followed by '90 00' 5. Command returns the data object consisting of the '7F49' template with the generated public key and modulus (RSA), or point (ECDSA) followed by '90 00' 6. Command returns the data object consisting of the '7F49' template with the generated public key and modulus (RSA), or point (ECDSA) followed by '90 00' 7. Command returns '6A 80' (Incorrect parameter command data field) 8. Command returns '6A 86' (Incorrect parameter in P1 or P2)
Post Condition(s)	1. The on card private key has changed to the new computed value.

C.3.2.2 Contactless Interface

Purpose	Validates that the GENERATE ASYMMETRIC KEY PAIR command cannot be issued through the contactless interface.
Reference(s)	1. SP 800-73-3 Part 2, Table 2 2. AS05.03
Precondition(s)	1. Record the existing contents of the public key data object 2. A valid PIV Card is placed within the reading range of the contactless reader 3. There exists a valid PC/SC connection between the test system and an instance of the contactless reader 4. No other contactless card is within the proximity of the reader
Test Scenario	1. Send SELECT card command with, • AID == A0 00 00 03 08 00 00 10 00 01 00 2. Perform steps 1 of test C.3.2.1
Expected Result(s)	1. The command returns the APT 2. Step 1 referred above returns '6A 81' – Function not supported
Post Condition(s)	NA

Appendix D—Test Reports

Following execution of each test class, test labs will prepare a summary report to document and communicate the result of all test cases belonging to that class. A new report shall be kept for each separate run against the same unit under test. Each test report shall be signed and dated by the tester, the lab representative and the authorized NIST personnel when needed, and accompany a cover letter to be sent to the vendor to inform the outcome of the test. Templates are provided below as samples and demonstrate the type of information to be included in the reports. Actual format of the report and the quantity of information contained may vary among labs.

D.1 PIV Client API Test Results Summary

HTML report	C:\Program Files\PIV Test Runner\.\api_tests\reports\DataObjectsRepresentationResultsSummary.html
PDF report	C:\Program Files\PIV Test Runner\.\api_tests\reports\DataObjectsRepresentationResultsSummary.pdf

Test Run No	Test Time	Test Date	
	Name/Description	Version	Date
Implementation Under Test (IUT)			

Test Case	Test Description	Result
B.1	pivConnect - Connection Test Assertions	
B.1.1	Valid Path Test Assertions	
B.1.1.1	Initiate Exclusive Connection	
B.1.1.2	Initiate Shared Connection	
B.1.2	Test Assertions for Error Conditions	
B.1.2.1	Malformed Connection Description	
B.1.2.1.1	Malformed connectionDescription - Shared connection true	
B.1.2.1.2	Malformed connectionDescription - Shared connection false	
B.1.2.2	Attempting to Share/Lock an Exclusive Connection	
B.1.2.2.1	Sharing an Exclusive Connection	
B.1.2.2.2	Locking an Exclusive Connection	
B.1.2.3	Attempting to Lock a Shared Connection	
B.1.2.4	Attempting to Open an Unsupported Connection	
B.1.2.4.1	Attempting to Open an Unsupported Connection - Shared connection true	
B.1.2.4.2	Attempting to Open an Unsupported Connection - Shared connection false	
B.2	pivDisconnect - Disconnection Test Assertions	
B.2.1	Valid Test Assertions	
B.2.1.1	Disconnect an Exclusive Connection	
B.2.1.2	Disconnect a Shared Connection	
B.2.2	Test Assertions for Error Cases	
B.2.2.1	Attempt Disconnect with Invalid Card Handle	
B.2.2.2	Disconnecting a previously disconnected Client Application	
B.3	pivSelectCardApplication	
B.3.1	Valid Test Assertions	

Test Case	Test Description	Result
B.3.1.1	Select a Card Application with a full AID	
B.3.1.2	Use a right truncated AID to Select a Card Application	
B.3.2	Test Assertions for Error Conditions	
B.3.2.1	Detect and handle an invalid cardHandle reference.	
B.3.2.2	Detect and handle an invalid applicationAID.	
B.4	pivLogIntoCardApplication	
B.4.1	Valid Test Assertions	
B.4.1.1	Log on to the Card Application	
B.4.2	Test Assertions for Error Conditions	
B.4.2.1	Attempt Logon with an invalid cardHandle.	
B.4.2.2	Attempt Logon with a malformed authenticator.	
B.4.2.3	Attempt Logon with invalid authenticator	
B.5	pivLogoutOfCardApplication	
B.5.1	Valid Test Assertions	
B.5.1.1	Log out of the Card Application	
B.5.1.2	Attempt to Log out of Card Application without logging in	
B.5.2	Test Assertions for Error Conditions	
B.5.2.1	Attempt Log out with Invalid Cardhandle	
B.6	pivGetData	
B.6.1	Valid Test Assertions	
B.6.1.1	Read an object from the card through the client application	
B.6.1.1.1	CCC –without logging in	
B.6.1.1.2	CHUID –without logging in	
B.6.1.1.3	X509 Certificate PIV Authentication –without logging in	
B.6.1.1.4	Security Object –without logging in	
B.6.1.1.5	X509 Certificate Card Authentication –without logging in	
B.6.1.1.6	X509 Certificate Digital Signature –without logging in	
B.6.1.1.7	X509 Certificate Key Management –without logging in	
B.6.1.1.8	Discovery Object – without logging in	
B.6.1.1.9	Key History Object – without logging in	
B.6.1.1.10	20 retired Certificates for Key Management	
B.6.1.2	Satisfying Security Conditions enables read of the objects	
B.6.1.2.1	Cardholder Fingerprints – after logging in	
B.6.1.2.2	Cardholder Facial Image – after logging in	

Test Case	Test Description	Result
B.6.1.2.3	Printed Information – after logging in	
B.6.1.3.4	Iris Image – after logging in	
B.6.2	Test Assertions for Error Conditions	
B.6.2.1	Handle an invalid cardHandle	
B.6.2.2	Handle an invalid Object Identifier	
B.6.2.3	The Client Application can handle missing data object	
B.6.2.4	Security Conditions are enforced for Secured Objects	
B.6.2.4.1	Cardholder Fingerprints – without logging in	
B.6.2.4.2	Cardholder Facial Image – without logging in	
B.6.2.4.3	Printed Information – without logging in	
B.7	pivPutData	
B.7.1	Valid Test Assertions	
B.7.1.1	Write data to an object on the card through the Client Application	
B.7.1.1.1	CCC – after logging in	
B.7.1.1.2	CHUID – after logging in	
B.7.1.1.3	X509 Certificate PIV Authentication – after logging in	
B.7.1.1.4	Cardholder Fingerprints – after logging in	
B.7.1.1.5	Security Object – after logging in	
B.7.1.1.6	X509 Certificate Card Authentication – after logging in	
B.7.1.1.7	X509 Certificate Digital Signature – after logging in	
B.7.1.1.8	Cardholder Facial Image – after logging in	
B.7.1.1.9	X509 Certificate Key Management – after logging in	
B.7.1.1.10	Printed Information – after logging in	
B.7.1.1.11	Discovery Object – after logging in	
B.7.1.1.12	Key History Object – after logging in	
B.7.1.1.13	20 retired X.509 Certificate for Key Management – after logging in	
B.7.1.1.14	Iris Image – after logging in	
B.7.2	Test Assertions for Error Conditions	
B.7.2.1	Identify and handle an invalid card handle	
B.7.2.2	Identify and handle an invalid Object Identifier (OID)	
B.7.2.3	Security Conditions are enforced for writing data to the data objects	
B.7.2.3.1	CCC – without logging in	
B.7.2.3.2	CHUID – without logging in	
B.7.2.3.3	X509 Certificate PIV Authentication – without logging in	

Test Case	Test Description	Result
B.7.2.3.4	Fingerprints – without logging in	
B.7.2.3.5	Security Object – without logging in	
B.7.2.3.6	X509 Certificate Card Authentication – without logging in	
B.7.2.3.7	X509 Certificate Digital Signature – without logging in	
B.7.2.3.8	Facial Image – without logging in	
B.7.2.3.9	X509 Certificate Key Management – without logging in	
B.7.2.3.10	Printed Information – without logging in	
B.7.2.3.11	Discovery Object – without logging in	
B.7.2.3.12	Key History Object – without logging in	
B.7.2.3.13	20 retired X.509 Certificate for Key Management – without logging in	
B.7.2.3.14	Iris Image – without logging in	
B.8	pivGenerateKeyPair	
B.8.1	Valid Test Assertions	
B.8.1.1	Generate an asymmetric key pair	
B.8.1.1.1	PIV Authentication key pair (9A)	
B.8.1.1.2	Digital Signature key pair (9C)	
B.8.1.1.3	Key Management key pair (9D)	
B.8.1.1.4	Card Authentication key pair (9E)	
B.8.2	Test Assertions for Error Conditions	
B.8.2.1	Identify and handle an invalid card handle	
B.8.2.2	Identify and handle an invalid keyReference	
B.8.2.3	Identify and handle an invalid cryptographicMechanism	
B.8.2.4	Security Conditions Not Satisfied	
B.9	pivCrypt	
B.9.1	Valid Test Assertions – security condition satisfied (PIN verified)	
B.9.1.1	AuthKey(9A) with Algorithm Identifier (xx)-Internal Authenticate	
B.9.1.2	AdminKey (9B) Algorithm Identifier (xx) - Mutual Authenticate	
B.9.1.3	Card Auth Key (9E) with Alg. ID (xx) External Authenticate	
B.9.1.4	Card Auth Key (9E symmetric) with Alg. ID (xx) Internal Authenticate	
B.9.1.5	Card Auth Key (9E asymmetric) with Alg. ID (xx) Internal Authenticate	
B.9.1.6	Digital Signature Key (9C) with Alg. ID (xx) Internal Authenticate (sign)	
B.9.1.7	Key Management Key (9D) and Alg. ID (xx) Internal Authenticate (transport/EC DH)	
B.9.2	Test Assertions for Error Conditions	

Test Case	Test Description	Result
B.9.2.1	Identify and handle invalid card handles	
B.9.2.2	Identify and handle invalid key references	
B.9.2.3	Identify and handle invalid input data	
B.9.2.4	Security conditions (PIN verified) not enforced	
B.9.2.4.1	AuthKey(9A) with Algorithm Identifier (xx)-Internal Authenticate (fail)	
B.9.2.4.2	Card Auth Key (9E symmetric) with Alg. ID (xx) Internal Authenticate (fail)	
B.9.2.4.3	Card Auth Key (9E asymmetric) with Alg. ID (xx) Internal Authenticate (success)	
B.9.2.4.4	Digital Signature Key (9C) with Alg. ID (xx) Internal Authenticate (sign) - fail	
B.9.2.4.5	Key Management Key (9D) and Alg. ID (xx) Internal Authenticate (transport) fail	

D.2 Card Command Interface Test Results Summary

HTML report	C:\Program Files\PIV Test Runner\.\reports\CardCommandInterfaceTestResultsSummary.html
PDF report	C:\Program Files\PIV Test Runner\.\reports\CardCommandInterfaceTestResultsSummary.pdf

Test Run No		Test Time		Test Date	
		Name/Description		Version	Date
Implementation Under Test (IUT)					

Test Case	Test Description	Result
C.1	Card Commands for Data Access	
C.1.1	SELECT Card Command	
C.1.1.1	Contact Interface	
	Long AID -Success	
	Short AID - Success	
C.1.1.2	Error Condition	
	Invalid AID	

Test Case	Test Description	Result
C.1.1.3	Contactless Interface	
	Long AID - Success	
	Short AID	
	Invalid AID	
C.1.2	GET DATA card command	
C.1.2.1	Contact Interface	
	NO PIN	
	CCC - no pin	
	CHUID - no pin	
	X509 PIV Authentication Certificate - no pin	
	Fingerprints - no pin	
	Printed Information - no pin	
	Facial Image - no pin	
	Iris Image – no pin	
	X509 Digital Signature Certificate - no pin	
	X509 Key Management Certificate- no pin	
	X509 Card Authentication Certificate- no pin	
	Security Object - no pin	
	Discovery Object – no pin	
	Key History Object – no pin	
	20 X509 retired Key Management Certificates – no pin	
	Iris Image data object – no pin	
	WITH PIN	
	Fingerprint - pin	
	Printed Information - pin	
	Facial Image - pin	
	Iris Image - pin	
	Unknown Object - Data Object Not Found	
C.1.2.2	Contactless Interface	
	NO PIN	
	CCC - no pin	

Test Case	Test Description	Result
	CHUID - no pin	
	X509 PIV Authentication Certificate - no pin	
	Fingerprints - no pin	
	Printed Information - no pin	
	Facial Image - no pin	
	Iris Image – no pin	
	X509 Digital Signature Certificate - no pin	
	X.509 Key Management - no pin	
	X509 Card Authentication Certificate- no pin	
	Security Object - no pin	
	Discovery Object – no pin	
	Key History Object – no pin	
	20 retired X.509 Key Management – no pin	
	Unknown Object - Data Object Not Found	
C.2	Commands For Authentication	
C.2.1	VERIFY Card Command	
C.2.1.1	Contact Interface	
C2.1.1.1	The PIV Card Application PIN only is supported (2a test case)	
	Valid PIN with invalid key reference - Key Reference Not Found	
	Valid PIN - Success	
	PIN with invalid format - Incorrect Parameter Data	
	Authentication method blocked	
	Blocked Card	
C2.1.1.2	The PIV Card Application PIN and Global PIN is supported (2B test case)	
	Valid PIN with invalid key reference - Key Reference Not Found	
	Valid PIN - Success	
	PIN with invalid format - Incorrect	

Test Case	Test Description	Result
	Parameter Data	
	Authentication method blocked	
	Blocked Card	
C.2.1.2	Contactless Interface	
C2.1.2.1	The PIV Card Application PIN only is supported (2a test case)	
	Valid PIN - Function Not Supported	
C2.1.2.2	The PIV Card Application PIN and Global PIN is supported	
	Valid PIN - Function Not Supported	
C.2.2	CHANGE REFERENCE DATA card command	
C.2.2.1	Contact Interface	
	Valid PIN with invalid key reference - Key Reference Not Found	
C.2.2.1.1	The PIV Card Application PIN only is supported (2a test case)	
	Valid PIN with invalid key reference 0x00 - Key Reference Not Found	
	Valid PIN – Success	
	PIN with invalid format - Incorrect Parameter Data	
	Incorrect PIN – x tries remaining	
	Incorrect PIN – Card Locked	
C2.2.1.2	The PIV Card Application PIN and Global PIN is supported	
	Global PIN - Valid PIN – Success	
	Global PIN - PIN with invalid format - Incorrect Parameter Data	
	Global PIN - Incorrect PIN – x tries remaining	
	Global PIN - Incorrect PIN – Card Locked	
	PIV PIN - Valid PIN – Success	
	PIV PIN - PIN with invalid format - Incorrect Parameter Data	
	PIV PIN - Incorrect PIN – x tries	

Test Case	Test Description	Result
	remaining	
	PIV PIN - Incorrect PIN – Card Locked	
C.2.2.2	Contactless Interface	
C.2.2.2.1	The PIV Card Application PIN only is supported	
	Valid PIN - Function Not Supported	
C2.2.2.2	The PIV Card Application PIN and Global PIN is supported	
	Global PIN: Valid PIN - Function Not Supported	
	PIV PIN: Valid PIN - Function Not Supported	
C.2.3	RESET RETRY COUNTER command	
C.2.3.1	Contact Interface	
	Invalid Key Reference, valid new PIN and valid PUK - Key Reference Not Found	
	Valid new PIN with invalid format - Incorrect Param Data	
	Valid new PIN and valid PUK combination - Success	
	Valid new PIN and invalid PUK – X tries remain	
	Valid new PIN and invalid PUK – card locked	
C.2.3.2	Contactless Interface	
	Valid PIN and PUK combination - Function Not Supported	
C.2.4	GENERAL AUTHENTICATE command	
C.2.4.1	Contact Interface	
	PIN security condition set	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9A) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE (symmetric) with Admin Key (9E) and Algorithm (xx) - Success	

Test Case	Test Description	Result
	MUTUAL AUTHENTICATE (symmetric) with Admin Key (9B) and Algorithm (xx) - Success	
	EXTERNAL AUTHENTICATE (symmetric with Card Auth Key (9E) and Algorithm (xx) – Security Condition satisfied	
	EXTERNAL AUTHENTICATE (symmetric with Admin Key (9B) and Algorithm (xx) – Security Condition satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9C) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9D) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9E) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE symmetric) with Key (9E) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE asymmetric) with 20 retired KMK (9D) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE - Invalid key reference or algorithm - Incorrect Param P1 P2	
	INTERNAL AUTHENTICATE with Key 9A - Invalid data in the data field - Incorrect Parameter Data	
	INTERNAL AUTHENTICATE with Key 9A - Security Status Not Satisfied	
	PIN security condition not set	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9A) and Algorithm (xx)-security status not satisfied	
	INTERNAL AUTHENTICATE	

Test Case	Test Description	Result
	(symmetric) with Admin Key (9B) and Algorithm (xx)-security status not satisfied	
	MUTUAL AUTHENTICATE (symmetric) with Admin Key (9B) and Algorithm (xx)-security status not satisfied	
	EXTERNAL AUTHENTICATE (symmetric with Card Auth Key (9E) and Algorithm (xx) – Security Condition satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9C) and Algorithm (xx)-security status not satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9D) and Algorithm (xx)-security status not satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9E) and Algorithm (xx) - Success	
	INTERNAL AUTHENTICATE symmetric) with Key (9E) and Algorithm (xx), - Success	
	INTERNAL AUTHENTICATE asymmetric) with 20 retired KMK (9D) and Algorithm (xx), - security status not satisfied	
C.2.4.2	Contactless Interface	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9A) and Algorithm (xx), - Security Condition not Satisfied	
	INTERNAL AUTHENTICATE (symmetric) with Admin Key (9B) and Algorithm (xx) - Security Condition not Satisfied	
	MUTUAL AUTHENTICATE (symmetric) with Admin Key (9B) and Algorithm (xx) - Security Condition not Satisfied	
	EXTERNAL AUTHENTICATE (symmetric with Card Auth Key (9E) and	

Test Case	Test Description	Result
	Algorithm (xx) – Security Condition Satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9C) and Algorithm (xx)-security status not satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9D) and Algorithm (xx)-security status not satisfied	
	INTERNAL AUTHENTICATE (asymmetric) with Key (9E) and Algorithm (xx) - Success	
	INTERNAL AUTHENTICATE symmetric) with Key (9E) and Algorithm (xx), - Success	
C.3	Card Commands for Credential Initialization and Administration	
C.3.1	PUT DATA Command	
C.3.1.1	Contact Interface	
	Without mutual authentication of PIV Card Application and the test toolkit application	
	CCC - no authentication - Security Status Not Satisfied	
	CHUID - no authentication - Security Status Not Satisfied	
	X509 PIV Authentication Certificate - no authentication - Security Status Not Satisfied	
	Fingerprint - no authentication - Security Status Not Satisfied	
	Printed Information - no authentication - Security Status Not Satisfied	
	Facial Image - no authentication - Security Status Not Satisfied	
	X509 Digital Signature Certificate - no authentication - Security Status Not Satisfied	
	X509 Key Management Certificate - no	

Test Case	Test Description	Result
	authentication - Security Status Not Satisfied	
	X509 Card Authentication Certificate - no authentication - Security Status Not Satisfied	
	Discovery Object no authentication - Security Status Not Satisfied	
	Security Object - no authentication - Security Status Not Satisfied	
	Key History Object - no authentication - Security Status Not Satisfied	
	Twenty retired X.509 Certificates for Key Management - no authentication - Security Status Not Satisfied	
	Iris Image Data Object - no authentication - Security Status Not Satisfied	
	With mutual authentication of the Card Application and Test Toolkit application	
	CCC - Security Status Satisfied- Success	
	CHUID -Security Status Not Satisfied - Success	
	X509 PIV Authentication Certificate - Security Status Satisfied - Success	
	Fingerprint -Security Status Satisfied - Success	
	Printed Information -Security Status Satisfied - Success	
	Facial Image - Security Status Satisfied - Success	
	X509 Digital Signature Certificate - Security Status Satisfied - Success	
	X509 Key Management Certificate - Security Status Satisfied - Success	
	X509 Card Authentication Certificate - Security Status Satisfied - Success	
	Discovery Object –Security Status Satisfied - Success	
	Security Object - Security Status Satisfied	

Test Case	Test Description	Result
	- Success	
	Key History Object - Security Status Satisfied - Success	
	Twenty retired X.509 Certificates for Key Management - Security Status Satisfied - Success	
	Iris Image Data Object - Security Status Satisfied - Success	
C.3.1.2	Contactless Interface	
	CHUID - Function Not Supported	
C.3.2	GENERATE ASYMMETRIC KEY PAIR command	
C.3.2.1	Contact Interface	
	PIV Authentication Key (9A) and Algorithm (xx) - Security Status Not Satisfied	
	PIV Authentication Key (9A) and Algorithm (xx), Security Status Satisfied - Success	
	Card Authentication Key (9E - asymmetric) and Algorithm (xx), Security Status Satisfied - Success	
	Digital Signature Key (9C) and Algorithm (xx), Security Status Satisfied - Success	
	Key Management Key (9D) and Algorithm (xx), Security Status Satisfied - Success	
	PIV Authentication Key (9A) and Algorithm (xx) - Invalid Cryptographic Mechanism Identifier - Incorrect Param Data field	
	Unknown Key reference or Algorithm - Invalid key reference - Incorrect Param P2	
C.3.2.2	Contactless Interface	
	Function Not Supported	

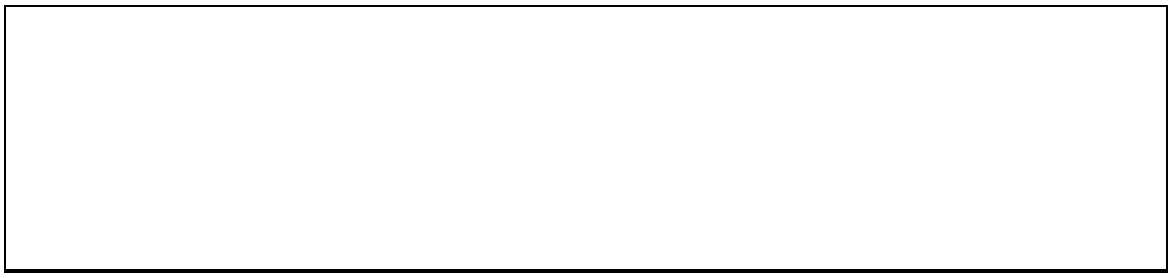
Test Case	Test Description	Result
C.4	CHECK tests[11]	
C.4.3	Checking Put Data for Correct Storage	
C.4.3.1	Contact Interface	
	CCC	
	CHUID	
	X509 Certificate PIV Authentication	
	Cardholder Fingerprint 1	
	Printed Information	
	Cardholder Facial Image	
	X509 Certificate Digital Signature	
	X509 Certificate Key Management	
	X509 Certificate Card Authentication	
	Security Object	
	Discovery Object	
C.4.4	Checking Application Property Template & Currently Selected Application	
C.4.4.1	Contact Interface	
	Application Property Template in Correct Format + Currently Selected Application is same	
C.4.5	Checking Retry Counter Auto Resets	
C.4.5.1	Contact Interface	
	Successful Verify command resets the retry counter	
	Successful Change Reference Data command resets the retry counter	
	Successful Reset Retry Counter command resets the retry counter	
C.4.6	Checking Retry Counter Countdown and Eventual Authentication Block	
C.4.6.1	Contact Interface	
	Verify APDU	
	Change Reference Data APDU	

[11] Tests under this sub-section are verification of post-conditions and state variables for test scenarios already described in Section C.1 through C.3.

Test Case	Test Description	Result
	Reset Retry Counter APDU	
C.4.7	Checking Change Reference Data Post-Condition	
C.4.7.1	Contact Interface	
	PIN successfully changed	
C.4.8	Checking Reset Retry Counter Post-Condition	
C.4.8.1	Contact Interface	
	PIN successfully changed	

D.3 Test Discrepancy Report

Test Case Information			
Name/Description		Number	
Conductor		Test Run	
Implementation Under Test (IUT)		Version	
Test Date		Time	
Type of Test	o Client API	o Card Command	o Data Object
Test Result	o PASS		o FAIL
Test Details			
Command / Function name			
Test Step No			
Expected Result			
Outcome			
Reason for Discrepancy			
Corrective steps			
Comments			



Appendix E—DTRs to Test Assertion Mapping

The following table provides an association between the Required Test Procedures in DTRs in Appendix A (those that can be electronically tested) and the test assertions in Appendix B and Appendix C.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE01.08.01	C.1.1.1 Test Scenario #1	The tester shall validate that there is a default selected card application which is the one specified by the vendor in VE01.08.01
TE02.03.01	Collectively by all tests in C.1.2 (Get Data) and C.3.1 (Put Data)	
TE03.01.01	Collectively by all tests in C.2.4 (General Authenticate) and C.3.2 (Generate Asymmetric Key Pair)	The tester shall validate the presence of all algorithm identifiers as indicated on the vendor documentation and the card, and that they comply with Tables 3.1, 5.1 and 6-2 of SP 800-78-2
TE03.02.02	C.1.1.1 Test Scenario #2	The tester shall validate that the information provided in VE03.02.01, is actually implemented by the card.
TE03.03.02	B.4.1.1 Log on to the Card Application B.4.2.1 Attempt Logon with an invalid cardHandle B.4.2.2 Attempt Logon with a malformed authenticator.	The tester shall validate that the card returns the correct tags and values in the authenticator data object as specified in Table 3, Part 3 of SP 800-73-3.
TE03.04.01	B.1.1.1 Initiate Exclusive Connection B.1.1.2 Initiate Shared Connection B.3.1.1 Select a Card Application with a full AID B.4.1.1 Log on to the Card Application B.5.1.1 Log out of the Card Application B.6.1.1 Get a reference to data object that does not require Login B.7.1.1 Write data to an object through the Client Application B.8.1.1 Generate an asymmetric key pair B.9.1.1 Authenticate the Card Application to Client Application	The tester shall validate the presence of the information provided in VE03.04.01 and that the Connection Description Template sent to the card conforms to Table 2, Part 3 of SP 800-73-3.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE03.08.01	C.2.3.1 RESET RETRY COUNTER command	The tester shall select the PIV Card Application and attempt to reset the Global PIN retry counter with the RESET RETRY COUNTER command and validate that the global PIN retry counter was not reset.
TE04.02.01	B.1.1.1 Initiate Exclusive Connection B.1.1.2 Initiate Shared Connection B.1.2.1 Malformed Connection Description B.1.2.2 Attempting to Share/Lock an Exclusive B.1.2.3 Attempting to Lock a Shared Connection B.1.2.4 Attempting to Open an Unsupported Connection	The tester shall validate that the client application implements the pivConnect as per SP 800-73-3 Part 3.
TE04.03.01	B.2.1.1 Disconnect an Exclusive Connection B.2.1.2 Disconnect a Shared Connection B.2.2.1 Attempt Disconnect with Invalid Card Handle B.2.2.2 Disconnecting a previously disconnected Client Application	The tester shall validate that the client application implements the pivDisconnect as per SP 800-73-3 Part 3.
TE04.04.01	B.3.1.1 Select a Card Application with a full AID B.3.1.2 Use a right truncated AID to Select a Card Application B.3.2.1 Detect and handle an invalid cardHandle reference B.3.2.2 Detect and handle an invalid applicationAID	The tester shall validate that the client application implements the pivSelectCardApplication as per SP 800-73-3 Part 3
TE04.05.01	B.4.1.1 Log on to the Card Application B.4.2.1 Attempt Logon with an invalid cardHandle B.4.2.2 Attempt Logon with a malformed authenticator B.4.2.3 Attempt Logon with invalid authenticator	The tester shall validate that the client application implements the pivLogIntoCardApplication as per SP 800-73-3 Part 3.
TE04.06.01	B.6.1.1 Get a reference to data object that does not require Login B.6.1.2 Get a reference to data object that requires Login B.6.2.1 Handle an invalid cardHandle B.6.2.2 Handle an invalid Object Identifier	The tester shall validate that the client application implements the pivGetData as per SP 800-73-3 Part 3.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
	B.6.2.3 The Client Application can handle missing data object B.6.2.4 Security Conditions are enforced for Secured Objects	
TE04.07.01	B.5.1.1 Log out of the Card Application B.5.2.1 Attempt Log out with Invalid Cardhandle	The tester shall validate that the client application implements the pivLogoutOfCardApplication as per SP 800-73-3 Part 3.
TE04.08.01	B.9.1.1 Encrypt a sequence of bytes B.9.1.2 Mutual Authentication of Client Application and Card Application B.9.2.1 Identify and handle invalid card handles B.9.2.2 Identify and handle invalid key references or algorithm B.9.2.3 Identify and handle invalid input data B.9.2.4 Security Conditions are Enforced	The tester shall validate that the client application implements the pivCrypt as per SP 800-73-3 Part 3.
TE04.09.01	B.7.1.1 Write data to an object on the Card through the Client Application B.7.2.1 Identify and handle an invalid card handle B.7.2.2 Identify and handle an invalid Object Identifier (OID) B.7.2.3 Security Conditions are enforced for writing data to the on-card data containers	The tester shall validate that the client application implements the pivPutData as per SP 800-73-3.
TE04.10.01	B.8.1.1 Generate an asymmetric key pair B.8.2.1 Identify and handle an invalid card handle B.8.2.2 Identify and handle an invalid keyReference or Key-Algorithm combination B.8.2.3 Identify and handle an invalid cryptographicMechanism B.8.2.4 Security Conditions are Enforced	The tester shall validate that the client application implements the pivGenerateKeyPair as per SP 800-73-3 Part 3.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE04.11.01	B10.1.1 Retrieve the supported PIV MiddlwareVersion	The tester shall validate that the client application supports all functions listed in table 1 of SP 800-73-3 and implements the pivMiddlewareVersion as per SP 800-73-3 Part 3 by returning the "800-73-3 Client API" parameter string
TE05.01.02	Collectively by all tests in Appendix C.	The tester shall validate that the card implements all the commands as required in Table 2, Part 2 of SP 800-73-3
TE05.01.03	Collectively by all tests in Appendix C.	The tester shall validate that the commands are implemented only through the interfaces allowed as shown in Table 2, Part 2 of SP 800-73-3
TE05.01.04	Collectively by all tests in Appendix C.	The tester shall validate that the commands are implemented only after the security condition associated with them are satisfied, as shown in the table, via the specified interface.
TE05.01.05	Collectively by all tests in Appendix C.	The tester shall validate that only the commands as indicated in the table are allowed for chaining via the interface supported after the security condition is satisfied.
TE05.05.01	C.1.1.1 Test Scenario #1 - Select card command - contact interface	The tester shall validate that the PIV Card Application is selected by providing its application identifier as specified in VE05.05.01. VE05.05.01: The vendor shall specify in its documentation the PIV Card Application Identifier.
TE05.07.02	C.1.1.1 Test Scenario #2 Select card command - contact interface	If the card implements the short version application selection, the tester shall validate that the PIV application is selectable by the right-truncated SELECT command.
TE05.09.01	C.1.1.1 Test Scenarios #1 and 2 in sequence - Select card command - contact interface	The tester shall validate that when the currently selected application is the PIV Card Application, when the SELECT APPLICATION command is sent with an AID that is either the AID of the PIV Card Application or its right-truncated version, then the PIV Card Application shall continue to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall be unchanged
TE05.10.01	C.1.1.2 Test Scenarios #1,2 and 3 in sequence - Select card command - Error Condition	TE05.10.01: The tester shall validate that when the currently selected application is the PIV Card Application, if the SELECT APPLICATION command is sent with an AID that is neither the AID of the PIV Card Application nor an AID supported by the ICC,

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
		the PIV Card Application continues to be the currently selected application and the setting of all security status indicators in the PIV Card Application shall be unchanged.
TE05.12A.01	Collectively by all test scenarios under C.1.2.1 Get Data card command - contact interface	For implementations without the Discovery Object or implementations with the Discovery Object implemented and the PIN usage policy's first byte se to 0x40, the Tester shall validate that all data objects that require the PIV Card Application PIN shall only be accessible after a successful validation of that PIN (through the VERIFY command).
TE05.12A.02	Collectively by test scenarios 13 under C.1.2.1 Get Data card command - contact interface Collectively by test scenario C.2.1.1 VERIFY card command – contact interface	In vendor products with the Discovery Object implemented, the Tester shall first check that the PIN usage policy's first byte is set to 0x60. The Tester shall next validate that all data objects that require a PIN shall be accessible after a successful validation of the Global PIN (in addition to the PIV Card Application PIN) through the VERIFY command.
TE05.12A.03	Collectively by all test scenarios under C.1.2.1 Get Data command - contact interface	The Tester shall validate that all data objects whose access rule is "Always Read " shall be accessible with or without the validation of the PIV Card Application PIN or Global PIN (if implemented as indicated in the Discovery Object).
TE05.13.01	Collectively by the test scenarios under C.2.1.1 Verify command - contact interface	The tester shall validate that the PIV Card Application PIN can be used for PIV data object access and command execution. The tester shall validate that when the Global PIN satisfies the PIV ACRs for PIV data object access and command execution with both PIV Card Application PIN and Global PIN then: 1) the discovery object is implemented with the PIN Usage Policy set to 0x60 zz where zz is set to either 0x10 or 0x20 and 2) the Global PIN can be used for PIV data object access and command execution.
TE05.17.01	C.2.1.1 Test Scenario # 2a, Step 6 or 2b, Step 6 – Verify Card Command – Contact Interface	The tester shall validate the PIV Card Application returns '69 83' in response to the VERIFY command, when the retry counter associated with the key reference is zero.
TE05.18.01	C.2.1.1 Test Scenario # 2 (branch 2a, Step 6 or branch 2b, Step 6 – Verify Card Command – Contact Interface	The tester shall validate that when the authentication data in the command data field does not satisfy the criteria in Section 2.4.3, SP 800-73-3 Part 2, the card command fails, and the PIV Card Application returns the status word '6A 80'

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE05.20.01	C.2.1.1 Test Scenario # 2 (branch 2a, Step 3 or branch 2b, Step 3 – Verify Card Command – Contact Interface	The tester shall validate that the retry counter associated with the key reference shall be set to the reset retry value (not decremented), when the VERIFY command succeeds.
TE05.21.01	C.2.1.1 Test includes this condition - Verify card command - contact interface	The tester shall validate that when the VERIFY command fails; the retry counter associated with the key reference is decremented by one.
TE05.22.01A	C.2.1.1 Test Scenario # 2 (2a branch step 5 or 2b branch step 5)- Verify card command - contact interface	The tester shall validate that the vendor documentation contains the information required in VE05.22.01A the card returns status word '6A 80', when the PIN information in the reference data field of the command is not padded to 8 bytes,
TE05.22.01B	C.2.1.1 Test Scenario # 2 (2a branch step 1 or 2b branch step 1)- Verify card command - contact interface	The tester shall validate that the vendor documentation contains the information required in VE05.22.01B and the card returns status word '6A 88', when the key reference is set to a value other than what is supported by the card.
TE05.23.01	C.2.2.1 Test Scenario # 2 (2a branch step 1 or 2b branch step 1) Change Reference Data card command - contact interface	The tester shall validate that when the current value of the retry counter associated with the key reference is zero, **the** PIV Card Application returns '69 83'
TE05.26.01	C.2.2.1 Test Scenario # 2 Change Reference Data card command - contact interface	The tester shall validate that the vendor documentation states the required information in VE05.26.01 and the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference when the command succeeds.
TE05.27.01	C.2.2.1 Test includes this condition - Change Reference Data card command - contact interface	The tester shall validate that the vendor documentation contains the information required in VE05.27.01 and the retry counter associated with the key reference shall be decremented by one if the card command fails.
TE05.28.01	C.2.2.1 Test Scenario # 4 - Change Reference Data card command - contact interface	The tester shall validate that the vendor documentation contains the information required in VE05.28.01 and the old PIN is not changed and the card returns status word '6A 80', when either of the PIN information in the reference data field of the command is not padded to 8 bytes,
TE05.28.01A	C.2.2.1 Test Scenario # 3 - Change Reference Data card command - contact interface	The tester shall validate that the vendor documentation contains the information required in VE05.28.01A and the old PIN is not changed and the card returns status word '6A 88', when the key reference is set to a value other than what is supported by the card.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE05.30.01	C.2.3.1 Test Scenario # 5 - Reset Retry Counter command - contact interface	The tester shall validate that the information requested in VE05.30.02 and VE05.30.01 are present in the vendor documentation. (NOTE: Testing this condition will leave the card unusable for further tests since the reset counter is zero).
TE05.31.01	C.2.3.1 Test Scenario # 4 - Reset Retry Counter command - contact interface	The tester shall validate that when the card command succeeds, the PIN's retry counter is set to the PIN's reset retry value specified in VE05.31.01, and the security status of the PIN's key reference is not changed. If the PUK's retry counter can be reset, the tester shall validate that the PUK's retry counter was reset to its initial reset retry value.
TE05.32.01	C.2.3.1 Test Scenario # 5 includes this condition - Reset Retry Counter command - contact interface	The tester shall validate that the information requested in VE05.32.01 is present in the vendor documentation, the security status of the PIN's key reference is set to FALSE, and the PUK's retry counter is decremented by one
TE05.33.01	C.2.3.1 Test Scenario # 3 - Reset Retry Counter command - contact interface	The tester shall validate that the vendor documentation includes the information required in VE05.33.01 and that when either the PUK or the PIN of the command does not satisfy the criteria in Section 2.4.3 Part 2 of SP 800-73-3, the PIN's retry counter is not reset and the card returns '6A 80.'
TE05.33.01A	C.2.3.1 Test Scenario # 2- Reset Retry Counter command - contact interface	The tester shall validate that the vendor documentation includes the information required in VE05.33.01A and that when the key reference value is other than what is supported by the card, the card returns '6A 88.'
TE05.34.01	C.2.4.1 Test Scenario # 1,2 - General Authenticate command - contact interface (Tests 7 pertaining to INTERNAL AUTHENTICATE also apply)	The tester shall validate that the GENERAL AUTHENTICATE command is implemented to authenticate the Card to the client application.
TE05.34.02	C.2.4.1 Test Scenario # 4a, 4b - General Authenticate command - contact interface (Test 7 pertaining to EXTERNAL AUTHENTICATE may also apply)	The tester shall validate that the GENERAL AUTHENTICATE command is implemented to authenticate the client application to the card.
TE05.34.03	C.2.4.1 Test Scenario # 4a, 4b - General Authenticate command - contact interface (Test 7 pertaining to MUTUAL AUTHENTICATE may also apply)	The tester shall validate that the GENERAL AUTHENTICATE command is implemented to mutually authenticate the Card to the client application and the client application to the card.

DTR from Appendix A	Test Scenario from Appendix B or C	DTR Description
TE05.34.04	C.2.4.1 General Authenticate, Test 5	If the 9C key is implemented, the tester shall validate that the GENERAL AUTHENTICATE command is implemented to realize signing functionality
TE05.34.05	C.2.4.1 General Authenticate, Test 6	If the 9D key is implemented, the tester shall validate that the GENERAL AUTHENTICATE command is implemented to support the RSA key transport or Elliptic Curve Diffie Hellman key establishment schemes specified in SP800-78.
TE05.36.01	C.2.4.1 Test Scenario # 8 - General Authenticate command - contact interface	TE05.36.01: The tester shall validate that the vendor documentation contains the information required in VE05.36.01 and the card returns status word '6A 86', when an invalid value of algorithm reference (P1) or key reference (P2) is sent to the card.
TE05.36.01A	C.2.4.1 Test Scenario # 9 - General Authenticate command - contact interface	TE05.36.01A: The tester shall validate that the vendor documentation contains the information required in VE05.36.01A and the card returns status word '6A 80', when an invalid value in data field of the command is sent to the card.
TE05.36.01B	C.2.4.1 Test Scenario # 11 and 12 - General Authenticate command - contact interface	TE05.36.01B: The tester shall validate that the vendor documentation contains the information required in VE05.36.01B and the card returns status word '69 82, whenever the command is used to authenticate the card to the client application without prior PIN verification for PIN protected keys (The Card Authentication Key is excluded from this requirement)
TE05.37.01	C.3.1.1 Put Data command - contact interface	The tester shall validate that the card complies with the PUT DATA command as defined in SP 800-73-3.
TE05.38.01	C.3.2.1 Generate Asymmetric Key Pair command - contact interface	The tester shall validate that the card implements the algorithms associated with identifiers specified as part of VE05.38.01 requirement and that the public key returned is formatted based on data object tags specified in Table 9, Part 2 of SP 800-73-3. VE05.38.01: The vendor shall specify in its documentation the cryptographic mechanism identifiers (from Table 4, Part 1 of SP 800-73-3) that have been implemented on the card.
TE05.40.01	C.3.2.1 Generate Asymmetric Key Pair command - contact interface	The tester shall validate that the initial contents of the public key data is replaced in full by the generated data, following a GENERATE ASYMMETRIC KEY PAIR command.

Appendix F—PIV Middleware Implementation Considerations (Informative)

This appendix presents some aspects of software design that are important in developing conformant implementations of the End-Point Client Application (also called PIV Middleware) Programming Interface described in Part 3 of SP 800-73-3. The design involves two major processes.

+ Design Step 1 — The semantics of the return codes for various commands in the interface must be properly understood in order to arrive at the set of execution scenarios under which each of them will occur.

+ Design Step 2 — The semantics of the parameters in the language-neutral function signatures must be properly understood in order to determine appropriate types or structures when implementation-language bindings are created for these function signatures.

To facilitate "Design Step 1", Appendix F.1 provides a mapping of return codes from the various components in the program call environment to the PIV Middleware return codes specified in Part 3 of SP 800-73-3. These components and their associated return codes are:

+ PIV Card Application – APDU Response Codes

+ Smart-Card Reader Driver – PC/SC Return Codes

+ Local Function Call Checks (also called API checks) – Programmer Defined

To obtain a comprehensive list of PIV Middleware return codes and their association with various functions, a matrix is given in Appendix F.2

To facilitate "Design Step 2", Appendix F.2 provides C-Language bindings for the PIV Middleware function signatures. The translation from numeric return codes in C-Language to the text-based PIV Middleware return codes is provided in Appendix F.3

F.1 PIV Middleware Return Code Mappings:

A set of mappings from the APDU response codes (coming from the PIV Card Application), local PC/SC driver response codes and the outcome of local API-level parameter validations to PIV Middleware return codes are given for each function in the following tables.

1. pivConnect (By design does not map to a card APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	Smart Card is positioned in the reader and the reader returns a handle	*SCARD_S_SUCCESS from PCSC*
PIV_CONNECTION_ DESCRIPTION_ MALFORMED	Contents of "connection Description" does not conform to Table 2, Part 3 of SP 800-73-3	Middleware specific -Local Validation of a parameter value or *SCARD_E_UNKNOWN_READER SCARD_E_DUPLICATE_READER*
PIV_CONNECTION_ LOCKED	(a)Shared Connection exists and an exclusive connection is requested (b)Exclusive Connection exists and a shared/an exclusive connection is requested	*SCARD_E_SHARING_ VIOLATION from PCSC*
PIV_CONNECTION_ FAILURE	(a) No card in the reader (b) A wrong connection mode is specified (say ISDN connection parameters for a PC/SC reader) (c)Scard service not started (one reason: reader not connected to local host)	*SCARD_E_NO_SMARTCARD SCARD_E_UNKNOWN_CARD SCARD_W_UNRESPONSIVE_CAR D SCARD_E_INVALID_ATR SCARD_E_NO_SERVICE SCARD_E_NO_READERS_AVAILA BLE SCARD_F_COM_ERROR*

2. pivDisconnect (By design does not map to a card APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	Function is called with the existing card handle value	*SCARD_S_SUCCESS from PCSC*
PIV_INVALID_CARD_ HANDLE	Function is called with: (a)Non-existing value of the card handle (or) (b)Successful Disconnect already called with this card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	An event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CARD* *SCARD_E_READER_UNAVAILABLE*

3. pivSelectCardApplication (maps to SELECT APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	(a) An existing card handle (and) (b) A valid Application AID are sent.	90 00 from Card
PIV_CARD_APPLICATION _NOT_FOUND	Application AID does not refer to an AID that the card supports	6A 82 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_ CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CA RD* *SCARD_E_READER_ UNAVAILABLE*

4. pivLogIntoCardApplication (maps to VERIFY APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	(a)An existing card handle is sent (and) (b) Authenticators well-formed	90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CARD* *SCARD_E_READER_ UNAVAILABLE*
PIV_AUTHENTICATOR_ MALFORMED	The authenticators do not carry the right lengths or tags as specified in Table 3, Part 3 of SP 800-73-3	Middleware specific - Local Validation of a parameter value (or) 6A 80 from Card (or) 6A 88 from Card
PIV_AUTHENTICATION_ FAILURE	The reference data in the authenticator is invalid	63 Cx from Card 69 83 from Card

5. pivGetData (maps to GET DATA APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	(a)An existing card handle is sent (and) (b) OID is valid	90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CARD* *SCARD_E_READER_ UNAVAILABLE*
PIV_INVALID_OID	The OID is not found in Table 2, Part 1 of SP 800-73-3	Middleware specific - Local Validation of a parameter value
PIV_DATA_OBJECT_NOT_ FOUND[5.1]	The OID refers to a data object not found on the card	6A 82 from Card
PIV_SECURITY_ CONDITIONS_NOT_ SATISFIED	Access to the card object denoted by OID is denied since security condition for its access is not satisfied.	69 82 from Card

5.1 (Note): This code cannot be realized if the underlying PIV Card implements all optional objects in the PIV Data Model.

6. pivLogoutOfCardApplication (maps to SELECT APDU with non-PIV AID)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	An existing card handle is sent	6A 82 or 90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CARD* *SCARD_E_READER_ UNAVAILABLE*

7. pivCrypt (maps to GENERAL AUTHENTICATE APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	An existing card handle is sent (and) all other parameters are valid	90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR SCARD_W_UNPOWERED_CARD SCARD_W_RESET_CARD SCARD_W_REMOVED_CARD SCARD_E_READER_ UNAVAILABLE*
PIV_INVALID_KEYREF_ OR_ALGORITHM	The algorithmIdentifier or the keyReference parameter value is not supported by the card.	6A 86 from the Card
PIV_INPUT_BYTES_ MALFORMED	Encoding of Tags and challenge/response strings do not correspond to specification in Table 6, Part 2 of SP 800-73-3.	6A 80 from the Card
PIV_SECURITY_ CONDITIONS_NOT_ SATISFIED	Internal Authentication is Performed without providing PIN	69 82 from the Card

8. pivPutData (maps to PUT DATA APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	(a)An existing card handle is sent (and) all other parameters are valid	90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR SCARD_W_UNPOWERED_CARD SCARD_W_RESET_CARD SCARD_W_REMOVED_CARD SCARD_E_READER_ UNAVAILABLE*
PIV_INVALID_OID	The value of OID supplied is not the one found in Table 2, Part 1 of SP 800-73-3.	Middleware specific - Local Validation of a parameter value
PIV_SECURITY_ CONDITIONS_NOT_ SATISFIED	No mutual or external authentication of PIV Card Administrator has been performed	69 82 from the Card
PIV_INSUFFICIENT_ CARD_RESOURCE	The card does not have enough memory to perform this operation with the given parameter values	6A 84 from the Card

9.pivGenerateKeyPair (maps to GENERATE ASYMMETRIC KEY PAIR APDU)

Return Code	Condition for Occurrence	Card App/*PCSC* Return Code
PIV_OK	(a)An existing card handle is sent (and) all other parameters are valid	90 00 from Card
PIV_INVALID_CARD_ HANDLE	Function is called with: Non-existing value of the card handle	*SCARD_E_INVALID_ HANDLE from PCSC*
PIV_CARD_READER_ ERROR	(a)Reader is not connected to the local host (b) No card in the reader (c) an event has occurred that prevents communication with the card	*SCARD_F_COMM_ERROR* *SCARD_W_UNPOWERED_CARD* *SCARD_W_RESET_CARD* *SCARD_W_REMOVED_CARD* *SCARD_E_READER_ UNAVAILABLE*
PIV_INVALID_KEY _OR_KEYALG _COMBINATION	Key Reference not supported by the card (or) Key Reference-Cryptographic Algorithm combination is not supported on the Card	6A 86 from the Card
PIV_UNSUPPORTED_ CRYPTOGRAPHIC_ MECHANISM	The cryptographic mechanism value is not one of the algorithm identifiers found in Table 6-2 of SP 800-78-2 or supported by the card.	6A 80 from the Card
PIV_SECURITY_ CONDITIONS_NOT_ SATISFIED	No mutual or external authentication of PIV Card Administrator has been performed	69 82 from the Card

10. pivMiddlewareVersion (The pivMiddlewareVersion call does not evoke a card command)

Return Code	Condition for Occurrence	*PCSC* Return Code
NA/	N/A	N/A

F.2 C-Language Bindings for PIV Middleware Interface Specification

```
/**
 *  \typedef typedef unsigned long PIV_RV
 *  Function return type
 *  \ingroup piv */
typedef unsigned long   PIV_RV;

/**
 *  \typedef typedef unsigned char PIV_Bool
 *  Boolean type
 *  \ingroup piv */
typedef unsigned char   PIV_Bool;

/**
 *  \typedef typedef unsigned char PIV_Byte
 *  Byte type
 *  \ingroup piv */
typedef unsigned char   PIV_Byte;

*  \typedef typedef unsigned long PIV_ULong32
 *  PIV redefinition of ANSI unsigned long type
 *  \ingroup piv */
typedef unsigned long   PIV_ULong32;

/**
 *  \typedef typedef unsigned long PIV_CARDHANDLE
 *  PIV redefinition of ANSI unsigned long type
 *  \ingroup piv */
typedef unsigned long   PIV_CARDHANDLE;

/**
 *  \typedef typedef unsigned long PIV_MIDDLEWAREVERSION
 *  PIV redefinition of ANSI unsigned long type
 *  \ingroup piv */
typedef unsigned long   PIV_MIDDLEWAREVERSION;


//PIV_RV pivConnect (PIV_Bool sharedConnection,
                     PIV_Byte *connectionDescription,
                     PIV_ULong32 *pCDLength,
                     PIV_CARDHANDLE *pCardHandle)
typedef PIV_RV (*pivConnect)(PIV_Bool, PIV_Byte *, PIV_ULong32 *,
PIV_CARDHANDLE *);

//PIV_RV pivDisconnect (PIV_CARDHANDLE cardHandle)
```

```
typedef PIV_RV (*pivDisconnect) (PIV_CARDHANDLE);

insert typedef for
//PIV_RV pivMiddlewareVersion(PIV_MIDDLEWAREVERSION middlewareVersion)
Typedef PIV_RV (*pivMiddlewareVersion) (PIV_MIDDLEWAREVERSION);

/*  Entry Points for Data Access */
//PIV_RV pivLogIntoCardApplication (PIV_CARDHANDLE cardHandle,
                                    PIV_Byte *authenticators,
                                    PIV_ULong32 pAuthLength)
typedef PIV_RV (*pivLogIntoCardApplication) (PIV_CARDHANDLE, PIV_Byte *,
PIV_ULong32);

//PIV_RV pivSelectCardApplication (PIV_CARDHANDLE cardHandle,
                                   PIV_Byte *applicationAID,
                                   PIV_ULong32 aidLength,
                                   PIV_Byte *applicationProperties,
                                   PIV_ULong32 *pAPLength)
typedef PIV_RV (*pivSelectCardApplication) (PIV_CARDHANDLE, PIV_Byte *,
PIV_ULong32, PIV_Byte *, PIV_ULong32 *);

//PIV_RV pivGetData (PIV_CARDHANDLE cardHandle,
                     PIV_Byte *OID,
                     PIV_ULong32 oidLength,
                     PIV_Byte *data,
                     PIV_ULong32 *pDataLength)
typedef PIV_RV (*pivGetData) (PIV_CARDHANDLE, PIV_Byte *, PIV_ULong32,
PIV_Byte *, PIV_ULong32 *);

//PIV_RV pivLogoutOfCardApplication(PIV_CARDHANDLE cardHandle)
typedef PIV_RV (*pivLogoutOfCardApplication) (PIV_CARDHANDLE);

/*  Entry Points for Cryptographic Operations */
//PIV_RV pivCrypt (PIV_CARDHANDLE cardHandle,
                   PIV_Byte algID,
                   PIV_Byte keyReference,
                   PIV_Byte *algInput,
                   PIV_ULong32 inputLength, PIV_Byte *algOutput,
                   PIV_ULong32 *pOutputLength)
typedef PIV_RV (*pivCrypt) (PIV_CARDHANDLE, PIV_Byte, PIV_Byte, PIV_Byte *,
PIV_ULong32, PIV_Byte *, PIV_ULong32 *);


/* Entry Points for Credential Initialization and Administration */
//PIV_RV pivPutData (PIV_CARDHANDLE cardHandle,
                     PIV_Byte *OID,
```

```
                    PIV_ULong32 oidLength,
                    PIV_Byte *data,
                    PIV_ULong32 dataLength)
typedef PIV_RV (*pivPutData) (PIV_CARDHANDLE, PIV_Byte *, PIV_ULong32,
PIV_Byte *, PIV_ULong32);

//PIV_RV pivGenerateKeyPair (PIV_CARDHANDLE cardHandle,
                             PIV_Byte keyReference,
                             PIV_Byte cryptographicMechanism,
                             PIV_Byte *publicKey,
                             PIV_ULong32 *pKeyLength)
typedef PIV_RV (*pivGenerateKeyPair) (PIV_CARDHANDLE, PIV_Byte, PIV_Byte,
PIV_Byte *, PIV_ULong32 *);
```

F.3 Translation from C-Language Numeric Return Codes to text-based PIV Middleware Return Codes

PIV Middleware Return Code	Numeric Value	pivConnect	pivDisconnect	pivSelectCardApplication	pivLogIntoCardApplication	pivGetData	pivLogoutOfCardApplication	pivCrypt	pivPutData	pivGenerateKeyPair	pivMiddlewareVersion
PIV_OK	0	X	X	X	X	X	X	X	X	X	X
PIV_CONNECTION_DESCRIPTION_MALFORMED	1	X									
PIV_CONNECTION_FAILURE	2	X									
PIV_CONNECTION_LOCKED	3	X									
PIV_INVALID_CARD_HANDLE	5		X	X	X	X	X	X	X	X	
PIV_CARD_APPLICATION_NOT_FOUND	6			X							
PIV_AUTHENTICATOR_MALFORMED	7				X						
PIV_AUTHENTICATION_FAILURE	8				X						
PIV_INVALID_OID	9					X			X		
PIV_DATA_OBJECT_NOT_FOUND	10					X					
PIV_SECURITY_CONDITITIONS_NOT_SATISFIED	11					X		X	X	X	
PIV_INVALID_KEYREF_OR_ALGORITHM[1]	13							X			
PIV_INSUFFICIENT_BUFFER[2]	14			X		X		X		X	
PIV_INPUT_BYTES_MALFORMED	15							X			
PIV_INSUFFICIENT_CARD_RESOURCE	16								X		
PIV_UNSUPPORTED_CRYPTOGRAPHIC_MECHANISM	19									X	
PIV_CARD_READER_ERROR	22		X	X	X	X	X	X	X	X	
PIV_INVALID_KEY_OR_KEYALG_COMBINATION[1]	13									X	

1. PIV_INVALID_KEY_OR_KEYALG_COMBINATION and PIV_INVALID_KEYREF_OR_ALGORITHM use the same numeric value.
2 .PIV Middleware communicating with T=0 protocol cards may receive either 61 xx or 90 00 as response codes. In this situation, the PIV Middleware may either issue a GET RESPONSE command on its own or pass the return code to the calling application. To allow for both these behaviors this code has been introduced. Also there are situations where the calling application may not exactly know how much buffer space to allocate for some return data such as fingerprint minutiae or facial image. The return code will then provide a means to send this information to the calling application. (NOTE: This return code is not part of the Return Code Mappings in Appendix F.1)

Appendix G—Acronyms

The following acronyms and abbreviations are used throughout this standard:

AID Application Identifier
APDU Application Protocol Data Unit
API Application Programming Interface
APT Application Property Template

BER-TLV Basic Encoding Rules Tag-Length-Value
BSP Biometric Service Provider

CCC Card Capability Container
CHUID Card Holder Unique Identifier
CSP Cryptographic Service Provider

DES-ECB Data Encryption Standard Electronic Code Book
DTR Derived Test Requirement

ECC Elliptic Curve Cryptography
ECDSA Elliptic Curve Digital Signature Algorithm
ECDH Elliptic Curve Diffie-Hellman

FIPS Federal Information Processing Standards
FISMA Federal Information Security Management Act

HSPD Homeland Security Presidential Directive

ICC Integrated Circuit Chip
IEC International Electrotechnical Commission
ISDN Integrated Services Digital Network
ISO International Organization for Standardization
ITL Information Technology Laboratory
IUT Implementation Under Test

NIST National Institute of Standards and Technology

OID Object IDentifier
OMB Office of Management and Budget

P1 First parameter of a card command
P2 Second parameter of a card command
PC Personal Computer
PIN Personal Identification Number
PIV Personal Identity Verification
PKCS Public-Key Cryptography Standards
PIX Proprietary Identifier eXtension
PUK PIN Unblocking Key

RID Registered application provider IDentifier

SP Special Publication
TRD Test Run Detail
TRS Test Results Summary

Appendix H—References

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

[2] HSPD 12, *Policy for a Common Identification Standard for Federal Employees and Contractors*, August 27, 2004.

[3] NIST Special Publication 800-73-3, Interfaces for Personal Identity Verification, Part 1: *End-Point PIV Card Application Namespace, PIV Data Model and Representation*, Part 2: *End-Point PIV Card Application Card Command Interface*, Part 3: *End-Point PIV Client Application Programming Interface*, Part 4: *The PIV Transitional Interfaces and Data Model Specification,* (February 2010)

[4] NIST Special Publication 800-76-1, *Biometric Data Specification for Personal Identity Verification*, January 2007. (See http://csrc.nist.gov)

[5] NIST Special Publication 800-78-2, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, February 2010. (See http://csrc.nist.gov)

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

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