DMDC

Card Technologies & Identification Systems Division

Evaluation of NIST SP 800-73 End State Reference Implementation

Version 1.1

October 2005

Revision History Page

Issue Date	Document Version	Modification By	Change Description
Sep 19, 2005	Version 0.1	Dave Wasmuth	Initial Paper
Sep 20, 2005	Version 0.2	Dave Wasmuth	Started integrating comments
Sep 22, 2005	Version 1.0	Dave Wasmuth	Finished integrating comments
Oct 20, 2005	Version 1.1	Dave Wasmuth	Integrated with the ACO's evaluation
Oct 21, 2005	Version 1.1	Bob van Spyk	Review

Table of Contents

Table of Figures & Tables

1 INTRODUCTION

This document characterizes and evaluates the NIST reference implementation for End State interfaces specified in Part 3 of SP 800-73, Interfaces for Personal Identity Verification (PIV). The objective is to identify possible uses, as well as interpretations of SP 800-73, that can have a bearing on our implementation choices.

The reference implementation is available for download as a *zipped* archive from the Personal Identity Verification (PIV) Project – Library (http://csrc.nist.gov/piv-project/FIPS201-Public-Comments.html) on the NIST web site.

1.1 Background

NIST SP 800-73 is one of several 800 series standards that specify the details associated with Federal Information Processing Standard (FIPS) 201, Personal Identity Verification for Federal Employees and Contractors. FIPS 201, in turn, was developed in response to Homeland Security Presidential Directive 12, mandating that all government agencies will support interoperable, secure and reliable forms of identification.

The reference implementation provides an additional resource for evaluating the standard PIV interfaces. It can help determine how specific issues have been resolved by NIST. It can also help shape the testing of PIV compliant cards, middleware and applications. It is implemented using other Smart Card standards, such as, ISO 7816, Global Platform and PC/SC that provide guidance for Smart Card storage, environment and communication infrastructures.

1.2 Audience

This document is intended for Card Technologies and Identity Systems Section personnel, standards, test, development and evaluation teams.

1.3 Assumptions

- The reference implementation is a complete representation of the API, data model and card edge of the SP 800-73 End-Point
- The reference implementation middleware can be used to test PIV applications developed in-house with different physical Smart Card solutions
- The reference implementation can provide valuable insight at the code level into the implementation of the PIV End-Point solution
- It will also be an indicator of the potential usefulness of a PIV Transitional reference implementation

1.4 References

- *PIV Middleware Reference Implementation User's Guide*, Version 1.1, June 25, 2005.
- *PIV Reference Implementation User's Guide*, Version 1.1, June 10, 2005.
- *NIST Special Publication 800-73, Interfaces for Personal Identity Verification*, April 2005

2 ARCHITECTURAL OVERVIEW

The downloaded package includes two major components, a Smart Card emulator and a middleware implementation.

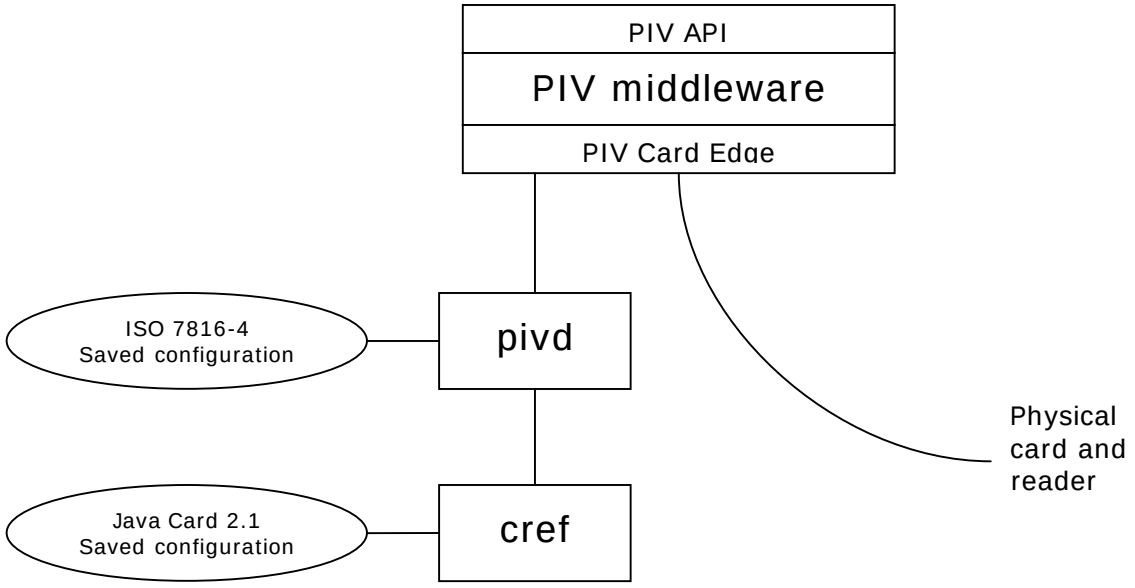
The card emulator implements the Global Platform commands LOAD and INSTALL without cryptography and includes a PIV application that implements the card edge commands specified in SP 800-73.

The middleware implementation provides the Application Programming Interface (API) specified in SP 800-73 and has a communication library that can be compiled to send card edge commands to the card emulator using the TLP-224 protocol or to a physical smart card through a PC/SC device. The card emulator is dependent upon these two external packages:

Crypto++ -- a free FIPS 140-2 compliant cryptography libraryJava Card™ 2.2.1 Developers' Kit

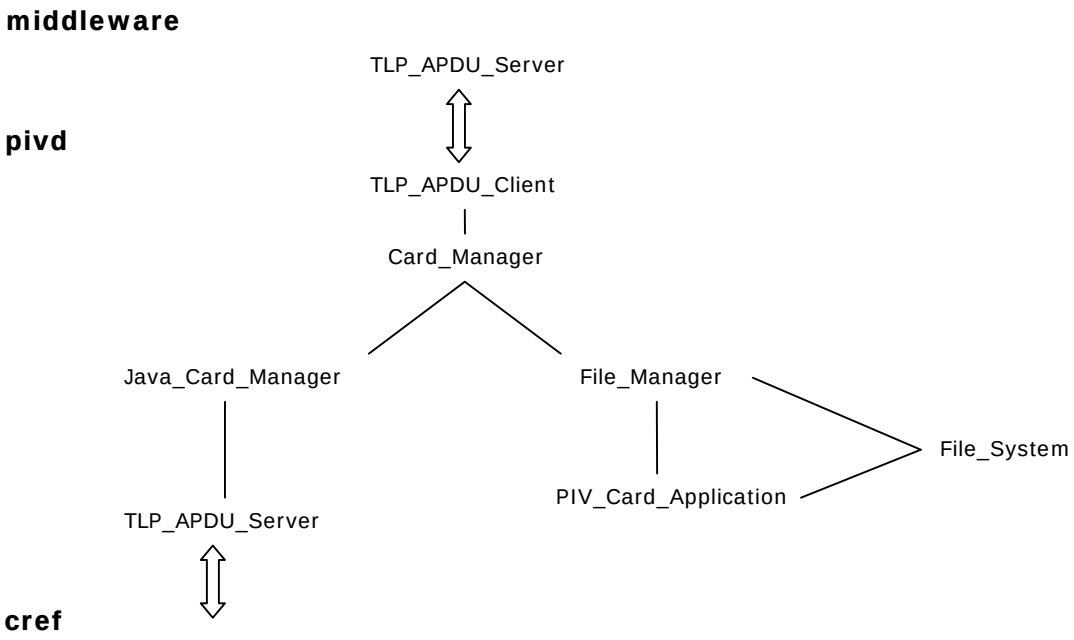
Figure 1 below depicts the high level architecture of the reference implementation. The card emulator program is called ***pivd*** and the Java Smart Card run time environment's program is called ***cref***. The PIV middleware package includes a character mode test UI that can be used to exercise each of the PIV Application Interface functions.

Figure 1. PIV Reference Implementation component architecture



The reference implementation components have a class structure that provides a modular design for sending, receiving and executing card edge APDU's. The major classes involved are depicted below in Figure 2.

Figure 2. Reference Implementation Class Structure



TLP_APDU_Server and TLP_APDU_Client handle the TLP-224 protocol communication between:

- An application using the PIV middleware API and the card emulator, **pivd**
- **pivd** and the Java Card 2.1 test program, **cref**

In **pivd**, the TLP_APDU_Client hands the APDU off to the Card_Manager which handles Global Platform's INSTALL and LOAD, as well as the PIV SELECT APDU's. Any others are passed along to the File_Manager or Java_Card_Manager, depending on which supports the currently selected application.

During initialization, the File_Manager instantiates the File_System object and loads it with hard-coded initial PIV values. It runs the PIV_Card_Application as a native card application, if it has been installed and loaded. When a PIV APDU is received by the File_Manager, the data are encrypted, and the new APDU is passed along to the PIV_Card_Application. The File_Manager and PIV_Card_Application both access the PIV data using the File_System object.

The Java_Card_Manager passes its ADPU's through to the Java Card test program, **cref**, for execution. The java applets running under **cref** have no access to the File_System object or, therefore, the PIV values managed by PIV_Card_Application.

3 ASSESSMENT

3.1 Evaluation

The reference implementation was compiled successfully. The Users' Guides supplied with the code is vague about how to load and execute applications in the **pivd** environment. Consequently, it has been tested with neither the card emulator nor a physical CAC or PIV card at this time.

3.2 Completeness

The reference implementation, along with its dependent packages, contains a set of components that provide support the development and testing of PIV client and card applications.

- Its middleware component implements all of the API calls and card edge commands specified in SP 800-73
- The middleware can be compiled to send card edge commands to either its card emulator environment or to a physical card environment
- It includes a PIV card application and some sample PIV data that can be loaded into the emulator to emulate a basic PIV compliant Smart Card
- Its integration with the Java Smart Card Developers' Kit allows for the deployment of custom applets for testing Java Card based PIV implementations

There are, however, some completeness issues. For example, the PIV functionality and data in the reference implementation are basic, the fingerprint and facial image buffers in the initial configuration are empty, as is the security object and there are no specific examples that use the PIV data apart from the data used to satisfy the API calls. Consequently, this evaluation concentrates on the End-Point Client API and Card Edge interface as specified in NIST SP 800-73, and not on a particular implementation of the PIV Card Application and PIV Data Model as described in FIPS-201.

End-Point Client API
The Client API is implemented in the source file `piv.c`. All of the methods from the specification are provided and compliant.

The `sequence of byte` argument type is implemented as two arguments, a byte array and a length. This is an expected implementation when using the C language.

End-Point Card Edge
The Card Edge interface is implemented in the source file `pivCardEdge.c`. All the methods are provided. Some minor differences were found from the specification, and are listed below:

- In GET DATA card command, the Lc is listed as a constant value in the specification but is variable in the implementation.
- In CHANGE REFERENCE DATA card command, P1 is listed as 0x00 in the specification but is implemented as 0x04.

Other Code Issues
The code has several "TODO comments" indicating that it is not really a fully tested and functioning product, which is to be expected. Here are some other examples of the reference implementation being less than a complete solution:

- A comment in File_Manager code states that it is hardwired to hand ADPU's off to the PIV_Card_Application for execution, although the User's Guide seems to indicate that applications can be added
- The File_Manager encrypts the data field of APDU's before passing them to the PIV_Card_Application, but just has a comment saying that the response "could be decrypted here"

3.3 Usability

Installation and Configuration

The documentation provided with the reference implementation provides a brief overview of the installation, contents and use of its two major components. It does not provide many specifics, however, which has led to various difficulties getting the components built and the environment up and running.

The installation and configuration of the middleware component of the reference implementation was straightforward. The issues with the Smart Card emulator component included the following:

- The provided documentation is incomplete and contains minor inconsistencies. Some of the inconsistencies include the following:
  - `PIVMW_HOME` is initially specified to include the code directory (which is required for compilation), but subsequent uses of `%PIVMW_HOME%` are only correct if this environment variable does not include the code subdirectory.
  - In order to compile crypto++ for use by the simulator, cryptlib's compiler options must be changed from multi-threaded to multi-threaded DLL (e.g for Debug change `/MTd` to `/MDd`)
  - Document states that `pivd.exe` is placed in `%PIV_HOME%\bin` by the build process, but this is actually placed in `%PIV_HOME%\build\bin`
- The console test application provided with the PIV Middleware Implementation is incomplete. This includes the following:
  - Hangs on selection of the 'q' (quit) command.
  - Does not currently implement commands for testing authentication or signing/decryption.
- The card simulator does not provide sample configuration files for loading the provided PIV Card Application implementation, nor does it detail the format of the configuration files.

Code Samples

The reference implementation has routines for BER-TLV encoding, is well organized and provides sample implementations of the SP 800-73 card edge commands, so it provides some useful examples. On the other hand, it is written in C++, rather than Java, is based on the ISO 7816-4 standard for file system cards, rather than the Java virtual card architecture, and is sparsely commented, which decreases the usefulness somewhat.

4 Utility of the Reference Implementation

The reference implementation environment may enable testing of a card application's integration with the End-Point middleware, connecting to either its card emulator or its physical PC/SC card support.

It may also be used during development, since its emulator component includes a PIV card application that provides a basic implementation of most of the PIV data model. (The facial and fingerprint images are empty and some issues were still unresolved when the reference implementation was packaged in June 2005.) Thus, newly developed card End-Point applets can be tested using the reference implementation components at an early stage in their development.

Lastly, the code samples may be useful when developing custom card implementations and PIV enabled middleware for the DoD's transitional solutions.

5 CONCLUSION

Although the reference implementation is incomplete, as indicated in this document, it can be of use for validating the interfaces specified in the SP 800-73 End-Point solution and possibly as an early test tool during PIV End-Point development.

Java Card development and testing takes advantage of a significant array of mature tools. The reference implementation will not replace any of those tools, but provides:

- Insight into NIST's interpretation of issues not fully specified in SP 800-73 End-Point.
- Insight into card applications and middleware at the code level
- One code level partial implementation model.
- Insight into specific uses of Java Card 2.1 vs C++

6 RECOMMENDATIONS

1. Test with live CAC and PIV cards to complete the review and assess its operational utility.
2. Acquire the 2004 NIST Transitional PIV reference implementation to augment, test, and document as an equivalent system supporting the Transitional solution in SP 800-73.