PKCS #11 v2.30: Cryptographic Token Interface Standard

RSA Laboratories

16 April 2009

Table of Contents

1 Introduction

As cryptography begins to see wide application and acceptance, one thing is increasingly clear: if it is going to be as effective as the underlying technology allows it to be, there must be interoperable standards. Even though vendors may agree on the basic cryptographic techniques, compatibility between implementations is by no means guaranteed. Interoperability requires strict adherence to agreed-upon standards.

Towards that goal, RSA Laboratories has developed, in cooperation with representatives of industry, academia and government, a family of standards called Public-Key Cryptography Standards, or PKCS for short.

PKCS is offered by RSA Laboratories to developers of computer systems employing public-key and related technology. It is RSA Laboratories' intention to improve and refine the standards in conjunction with computer system developers, with the goal of producing standards that most if not all developers adopt.

The role of RSA Laboratories in the standards-making process is four-fold:

1. Publish carefully written documents describing the standards.

2. Solicit opinions and advice from developers and users on useful or necessary changes and extensions.

3. Publish revised standards when appropriate.

4. Provide implementation guides and/or reference implementations.

During the process of PKCS development, RSA Laboratories retains final authority on each document, though input from reviewers is clearly influential. However, RSA Laboratories’ goal is to accelerate the development of formal standards, not to compete with such work. Thus, when a PKCS document is accepted as a base document for a formal standard, RSA Laboratories relinquishes its “ownership” of the document, giving way to the open standards development process. RSA Laboratories may continue to develop related documents, of course, under the terms described above.

PKCS documents and information are available online at `http://www.rsasecurity.com/rsalabs/PKCS/`. There is an electronic mailing list, “`cryptoki`”, at `rsasecurity.com`, specifically for discussion and development of PKCS #11. To subscribe to this list, send e-mail to `majordomo@rsasecurity.com` with the line “`subscribe cryptoki`” in the message body. To unsubscribe, send e-mail to `majordomo@rsasecurity.com` with the line “`unsubscribe cryptoki`” in the message body.

Comments on the PKCS documents, requests to register extensions to the standards, and suggestions for additional standards are welcomed. Address correspondence to:

PKCS Editor
RSA Laboratories
174 Middlesex Turnpike
Bedford, MA 01730 USA
pkcs-editor@rsasecurity.com
http://www.rsasecurity.com/rsalabs/PKCS/

It would be difficult to enumerate all the people and organizations who helped to produce PKCS #11. RSA Laboratories is grateful to each and every one of them. Special thanks go to Bruno Couillard of Chrysalis-ITS and John Centafont of NSA for the many hours they spent writing up parts of this document. Thanks also for the many other technical descriptions provided by many industry specialists. The reviewers of the document, without whose help the quality of the content would not be as great, must also be acknowledged and thanked. The review effort cannot be underestimated especially for a document so large.

Revision History

Version 1.0, PKCS #11's document editor was Aram Pérez of International Computer Services, under contract to RSA Laboratories; the project coordinator was Burt Kaliski of RSA Laboratories.

Version 2.01, Ray Sidney served as document editor and project coordinator.

Version 2.10 and Version 2.11, Matthew Wood of Intel was document editor and project coordinator.

Version 2.20, Simon McMahon from Eracom was editor for while Magnus Nystrom of RSA coordinated the project.

Version 2.30, Editor Simon McMahon with Robert Griffin of RSA as project coordinator.

2 Scope

This standard specifies an application programming interface (API), called “Cryptoki,” to devices which hold cryptographic information and perform cryptographic functions. Cryptoki, pronounced “crypto-key” and short for “cryptographic token interface,” follows a simple object-based approach, addressing the goals of technology independence (any kind of device) and resource sharing (multiple applications accessing multiple devices), presenting to applications a common, logical view of the device called a “cryptographic token”.

This document specifies the data types and functions available to an application requiring cryptographic services using the ANSI C programming language. These data types and functions will typically be provided via C header files by the supplier of a Cryptoki library. Generic ANSI C header files for Cryptoki are available from the PKCS Web page. This document and up-to-date errata for Cryptoki will also be available from the same place.

Additional documents may provide a generic, language-independent Cryptoki interface and/or bindings between Cryptoki and other programming languages.

Cryptoki isolates an application from the details of the cryptographic device. The application does not have to change to interface to a different type of device or to run in a different environment; thus, the application is portable. How Cryptoki provides this isolation is beyond the scope of this document, although some conventions for the support of multiple types of device will be addressed here and possibly in a separate document.

A number of cryptographic mechanisms (algorithms) are supported in this version. In addition, new mechanisms can be added later without changing the general interface. It is possible that additional mechanisms will be published from time to time in separate documents; it is also possible for token vendors to define their own mechanisms (although, for the sake of interoperability, registration through the PKCS process is preferable).

Cryptoki is intended for cryptographic devices associated with a single user, so some features that might be included in a general-purpose interface are omitted. For example, Cryptoki does not have a means of distinguishing multiple users. The focus is on a single user’s keys and perhaps a small number of certificates related to them. Moreover, the emphasis is on cryptography. While the device may perform useful non-cryptographic functions, such functions are left to other interfaces.

3 Document structure

PKCS#11	pkcs-11v2-30.pdf
PKCS#11 Base Functionality	pkcs-11v2-30b.pdf
PKCS#11 Mechanisms	pkcs-11v2-30m1.pdf
PKCS#11 Other Mechanisms	pkcs-11v2-30m2.pdf

A Intellectual property considerations

The RSA public-key cryptosystem is described in U.S. Patent 4,405,829, which expired on September 20, 2000. The RC5 block cipher is protected by U.S. Patents 5,724,428 and 5,835,600. RSA Security Inc. makes no other patent claims on the constructions described in this document, although specific underlying techniques may be covered.

RSA, RC2 and RC4 are registered trademarks of RSA Security Inc. RC5 is a trademark of RSA Security Inc.

CAST, CAST3, CAST5, and CAST128 are registered trademarks of Entrust Technologies. OS/2 and CDMF (Commercial Data Masking Facility) are registered trademarks of International Business Machines Corporation. LYNKS is a registered trademark of SPYRUS Corporation. IDEA is a registered trademark of Ascom Systec. Windows, Windows 3.1, Windows 95, Windows NT, and Developer Studio are registered trademarks of Microsoft Corporation. UNIX is a registered trademark of UNIX System Laboratories. FORTEZZA is a registered trademark of the National Security Agency.

License to copy this document is granted provided that it is identified as “RSA Security Inc. Public-Key Cryptography Standards (PKCS)” in all material mentioning or referencing this document.

RSA Security Inc. makes no other representations regarding intellectual property claims by other parties. Such determination is the responsibility of the user.

Revision History

This is the initial version of PKCS #11 v2.30.

Derived from PKCS#11 v2.20 where all of the standard was published as a single document.

This revision includes the major editorial change of splitting the document as well as significant new technical additions.