TECHNICAL SPECIFICATION

ISO/IEC TS 23465-2

First edition
2023-02

Card and security devices for personal identification — Programming interface for security devices —

Part 2: API definition

Cartes et dispositifs de sécurité pour l'identification personnelle — L'interface du logiciel pour dispositifs de sécurité —

Partie 2: Definition de API

Reference number
ISO/IEC TS 23465-2:2023(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see https://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 17, *Cards and security devices for personal identification*.

A list of all parts in the ISO/IEC 23465 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

Integrated chip card (ICC) technologies and solutions are widely deployed around the world, but the system for identity tokens and credentials is quickly changing. In this context, the application protocol data unit (APDU) protocol outlined in the ISO/IEC 7816 series is becoming in some cases a hindrance to the integration of ICs in environments such as mobile phones, handheld devices, connected devices (e.g. M2M, IoT) or other applications using security devices.

In addition, several stakeholders are not familiar with, or not very fond of the APDU protocol because of its complexity. They would circumvent its constraints by requesting an abstraction layer hiding IC specifics such as data structures and complexity of the security policies.

A common way to reach this goal in the software development is the definition and application of application programming interface (API) functions to access the IC within the devices. Specific knowledge of ADPU protocols and details of the IC implementation is not necessary anymore. Also, the complexity and details of the implementation of the security model and the security policy can be shifted from the pure application development into the system design of the whole ID management.

However, even solutions based on those kinds of middleware are perceived as cumbersome in some systems. The market looks for a middleware memory footprint to be as low as possible and the acceptance, usage and maintenance of such a system can be simpler.

This document aims to overcome or mitigate those issues by proposing a new approach that preserves ICC functionality and allows a seamless ICC portability onto new systems.

The ISO/IEC 23465 series focuses on a solution by designing an API and a system with the following characteristics:

— It offers a set of API calls related to multi-sectorial ICC functionality, derived from the ISO/IEC 7816 series of other ICC related standards.

— It defines the sub-system to perform the conversion from the API function to the interface of the security device (e.g. APDU-interface), called “proxy”.

— It results in a description of solutions with no middleware or very little middleware memory footprint (i.e. simplified drivers).

— It defines simplified ICC capabilities, description of the discoverability (i.e. with significantly less complexity than ISO/IEC 24727) and provides examples of usages.

The present model is static and future revisions are expected to add live cycle functionality.

Card and security devices for personal identification — Programming interface for security devices —

Part 2: API definition

1 Scope

This document describes the following aspects of the programming interface between the client application dealing with the security device and the proxy, based on the framework outlined in ISO/IEC 23465-1:

— the generic API definition;

— state and security models for use cases;

— class and API definitions of functionality, defined in other standards, e.g. the ISO/IEC 7816 series.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 23465-1:2023, *Card and security devices for personal identification — Programming interface for security devices — Part 1: Introduction and architecture description*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 23465-1 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1
array
indexed list of any data types with a well-known number of members

3.2
boolean
data type used to denote a data item that can only take one of the values TRUE and FALSE

3.3
char
data type containing an 8-bit quantity that encodes a single-byte character from any byte-oriented code set with a numerical value between 0 and 255

3.4
credential
set of data presented as evidence of a claimed or asserted identity and/or entitlements

EXAMPLE A user attribute (see ISO/IEC 19286) signed by the issuer as proof of authenticity is a credential that can be verified by the service provider by validating the electronic signature.

[SOURCE: ISO/IEC 29115:2013, 3.8, modified — Note 1 to entry was deleted. An EXAMPLE was added.]

3.5
integer
data type containing a sequence of digits taken from a number base

Note 1 to entry: Programming languages support integer values as a data type in different flavours, e.g. as signed integer or unsigned integer and in short or long format. To be programming language agnostic this document does not specify any of these different definitions and uses the general type integer for different types. This approach is different to the used interface description language (IDL).[6] It is the responsibility of the application programmer to define the type of integer in a relevant API function call according to the need of the function and the programming language used.

EXAMPLE Digits from the number base 10 (decimal) consisting of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

3.6
octet
data type with 8-bit quantity

3.7
string
data type containing a sequence of characters with a definite length

4 Symbols and abbreviated terms

APDU	application protocol data unit
API	application programming interface
CPLC	Card Production Life Cycle
CPS	Cryptographic Service Provider
DLOA	Digital Letter of Approval
eID	electronic identity
eSE	embedded secure element
eUICC	embedded universal integrated chip card
GP	GlobalPlatform
ID	identification
IDL	interface description language
KMS	Key Management System
OMG	Object Management Group
OS	operating system
OSI	open systems interconnection
PII	personal identifiable information
PIV	Personal Identity Verification
PKCS	Public Key Cryptographic Standard
SD	secure digital (memory card)
TSM	Trusted Service Manager eSE

5 Graduated APIs for security devices

5.1 General

A security device within an electronic device is characterized by its functionality.

A security device may act as

— a means for secure storage of credentials, without any additional functionality other than to retrieve the credentials,

— a means with cryptographic capabilities possibly storing credentials and offers in addition to cryptographic operations with these credentials,

— an eSE-application supporting device, storing the PII and offers eID-application related functionality. This may include related cryptographic capabilities and/or secure storage capabilities for any type of credentials.

Depending on the level of the use cases, different usage models and functionalities shall be considered. This leads to definitions of sub-sets of the full-flavoured APIs.

Figure 1 depicts the situation of an eSE-application supporting security device including the capabilities of a cryptography supporting device with means of a secure storage.

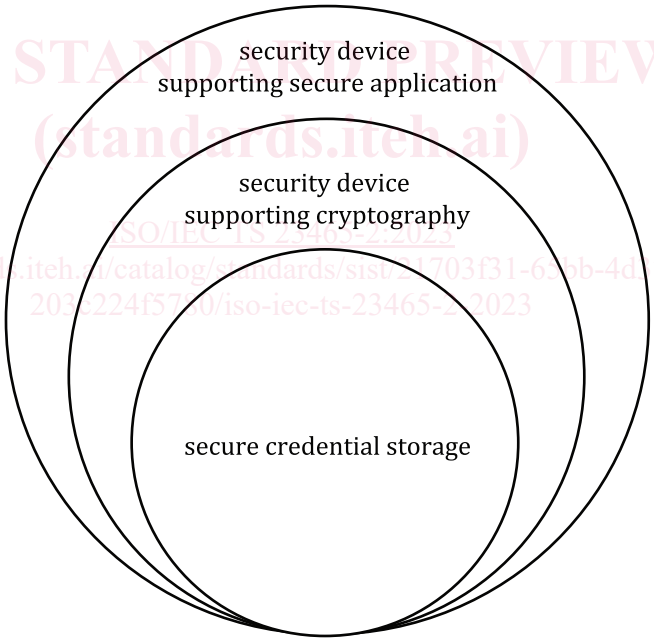

Figure 1 — Possible incorporated security device variants

5.2 Secure credential storage

In this use case the API functionalities are focused on the handling of the credentials. The administration, personalisation and usage of credentials are functions of this reduced API:

— Set data (see 10.4.5.5);

— Get data (see 10.4.5.4).

5.3 Security device supporting cryptography

A security device supporting cryptographic functionality supports a client application in addition to secure credential storage with additional security functionality related to cryptographic operations. There are a lot of cryptographic functions available, which allows an application to perform extended security protocols with the support of the security device. An example for such a cryptographic function is PKCS#11 functionality. [5]

A cryptographic supporting security device normally includes the creation, retrieval and deletion of credentials and is, therefore, also a credential storage.

Functions of security device supporting cryptography are e.g.:

— generate a Key (see 10.4.10.2);

— encrypt (see 10.5.3.2.2);

— decrypt (see 10.5.3.2.4).

5.4 Security device supporting secure application

Some client applications use a security device as a separate and secured application storage. An (ID-) application running on a standalone personal device, e.g. a health card or an ID document can be divided between a secured and unsecured part and be distributed, e.g. in a mobile application. For example, the secured part is located in the security device of the mobile device, the unsecured part is running in the mobile application itself. The combination of both the secured part and the unsecured part are completely running on the mobile device and can be a fully personalized ID-application communicating with a terminal.

A security device supporting ID applications may incorporate, in addition, the functionality of a cryptographic and/or secure storage supporting security device. Examples of functionality of such a security device are:

— PIV application;

— health application;

— mobile driving license.

6 API pre-requisite

6.1 Description language

The APIs defined in this document are outlined and defined with a generic interface description language (IDL). The IDL provides means to describe these interfaces in a language independent manner. Usually, additional language binding appendices are included in the basic descriptions of the IDL outlining how the IDL can be applied in the given language. The interface definition language defined in[6] is applied in this document. Additional information about the IDL is outlined in the informative Annex C.

6.2 Format of an API function

6.2.1 General

A generic API function consists of an explanatory name of the method/function, a list of parameters arguments and response data/values from the method invoke/function call. The name of a method/function, i.e. the API name, is typically understandable and self-explanatory. The terminology also signals the intent of the function. In this document the naming follows the java convention outlined in[3] and[4].

6.2.2 Addressing means

Addressing means allow the client application to use and address any security device on board of the electronic system.

6.2.3 Parameters

Most of method invokes/function calls need their related parameters. The parameters are a sequence of separated variables and shall be defined in each API separately.

6.2.4 Return values

The result of a method/function is returned to the invoker/caller. The type of the return value is method/function related and shall be defined for each API function separately.

The general format of an API method/function looks like:

`<API_Return_Value> <API_Function_Name> (<Parameters>…)`

`raises Exeception 1, Exception 2, ….`

The description of the API uses the following skeleton, defined in Table 1:

Table 1 — Format of the ISO/IEC 23465 API description

API Name	API_Function_Name
API Return value	***any*** API_Return_Value
API Parameter(s)	***in any*** inputparameter1,
	in any inputparameter2,
	…
	inout any inoutputparameter1
	inout any inoutputparameter2
	…
	out any outputparameter1
	out any outputparameter2
	…
Exceptions	Exception1
	Exception2
	…

The API function names standardized in this document use the prefix **`isoIec23465_`**. The case sensitivity follows the rules in conformance with the coding conventions in[3] and[4].

6.2.5 Callback functionality

To allow a non-blocking programming style or un-performant processing, the callback mechanism can apply. Synchronous and ansynchronous program processing are supported by many programming languages. These callback mechanisms may optionally be used.

If this applies, a callback function reference shall be conveyed in the parameter of each method. In the description tables of the methods in Clause 10, the callback function references in the parameter lists are not shown but have to be added if callback is used.

7 API error handling

7.1 General

Security devices using the ISO/IEC 7816 series APDUs indicate the processing status with the responses trailer SW1- SW2, especially for error conditions. In case of the API usage, the more efficient exception handling applies which is offered by modern programming languages.

The communication with the security device is performed by the API and its implementation. Any commands and responses to and from the security device are hidden from the application. In case of ISO/IEC 7816 related security devices, the response trailers are mapped to corresponding exceptions by the proxy.

7.2 Exceptions

The possible exceptions for each API method/function are outlined and explained in the API definition of each function.

Exceptions which are not dedicated to a specific method and are thrown, e.g. by the runtime environment or other components of the software system, are not listed explicitly. An example can be the exception “FunctionNotImplementedExeception”.

8 Security device identification

8.1 Security device attributes

As outlined in ISO/IEC 23465-1, this series of standards applies to systems where applications make use of security devices. It is possible that the system has access to more than one security device. A specific security device is characterized by a set of attributes. These attributes reflect high level information about the security device and shall allow a calling instance to identify the assigned security device. A set of attributes is proposed in Table 2.

Table 2 — Security device attributes

Type of information	Security device attribute	Type	Aim of information / data	Comment
Security device information	SecuritydeviceID	octet [ ]	Unique ID of the security device, Identifier of object of type SecurityDevice (see 10.4.3). can be absent or occur once or more times	Identification number, e.g. Card identification number For privacy reasons this ID can be replaced with a random number
	Security device capability_profile_ID	octet [ ]	Comprehensive description and abstraction of the capabilities of the security device and optionally the device, containing all information below	Profile can be used alternatively to describe SE (online availability), URL or OID can be possible
	Security device type	integer	Type of the security device	The types have to be defined in a separate table
	Security device owner	octet [ ]	Identifier of owner or issuer of the SE, also integrated-security-device owner	IIN Issuer Identification number
	Security device certification	octet [ ]	Information about certification of chip, OS and application (high level)	Data structure, e.g. GlobalPlatform DLOA
	Open Mobile API support	octet [ ]	Information regarding the support of OMAPI	Array of octet
	NFC support	boolean	Information regarding support of NFC	Yes/no
	CSP certification	octet [ ]	Information about certification of Cryptographic Service Provider CSP	Data structure including version and scheme
Chip information	Chip certification	octet [ ]	Information about the certification of the chip if other high-level information is not available	Array of octet
	Available chip memory	integer	Usable memory on the chip. In GP, this information is optional, it shall become mandatory.	Number of bytes on the security device
OS information	Available security device memory	integer	Usable memory in the OS.	Number of bytes in the OS
	Security device OS type, OS version, GP version and support	octet [ ]	OS-type and -version, GP-support and -version	TLV structure Java Card/native OS, OS-specification. GP specification, including version number
	SE_OS_Cryptography	octet [ ]	List of supported cryptographic algorithms/protocols	Bit map
GP information	Card Recognition Data	octet [ ]	Structure defined by GP-Services	TLV structure, see [8]
	Card Capability Data	octet [ ]	Structure defined by GP-Services	TLV structure, see [8]
Security device support	Key-Management-System information	octet [ ]	Information about the used key management system	relevant for the electronic system
	Discretionary Data	octet [ ]	Any data defined by the issuer	Array of octet

The security device attributes contain information derived from the different available security devices and additional data from the electronic system. The proxy collects the security device attribute from a security device and exposes it at the API (see Table 3). Security device information may also retrieved by usage of the Open Mobile API which is outlined in informative Annexes A and B.

8.2 Security device entry

In complex systems, an application has the choice to use at least one of several security devices. The securityDeviceEntry is a structure consisting of the securityDeviceNumber as a unique identification and numbering element and the descriptive attributes dedicated to the security device (see Table 2). The securityDeviceNumber assigned by the proxy, allows the selection of the security device for the further usage within the application. A securityDeviceEntry is defined as structure consisting of security device ID and the associated SdAttribute, mentioned in Table 3.

Table 3 — securityDeviceEntry

```
struct securityDeviceEntry {
                integer securityDeviceNumber;
                octet securityDeviceAttribute [ ];
        }
```

Subclause 10.3.1 describes a generic API function to get the list of available Security device entries in the electronic system.

9 Data model definition

9.1 General

The API used by a client software acts on instances of objects related to the security device system. In general, the API functions are methods and functions dealing with, and are assigned to, these objects. The general data model of a security device application is outlined in the class diagram in Figure 2.

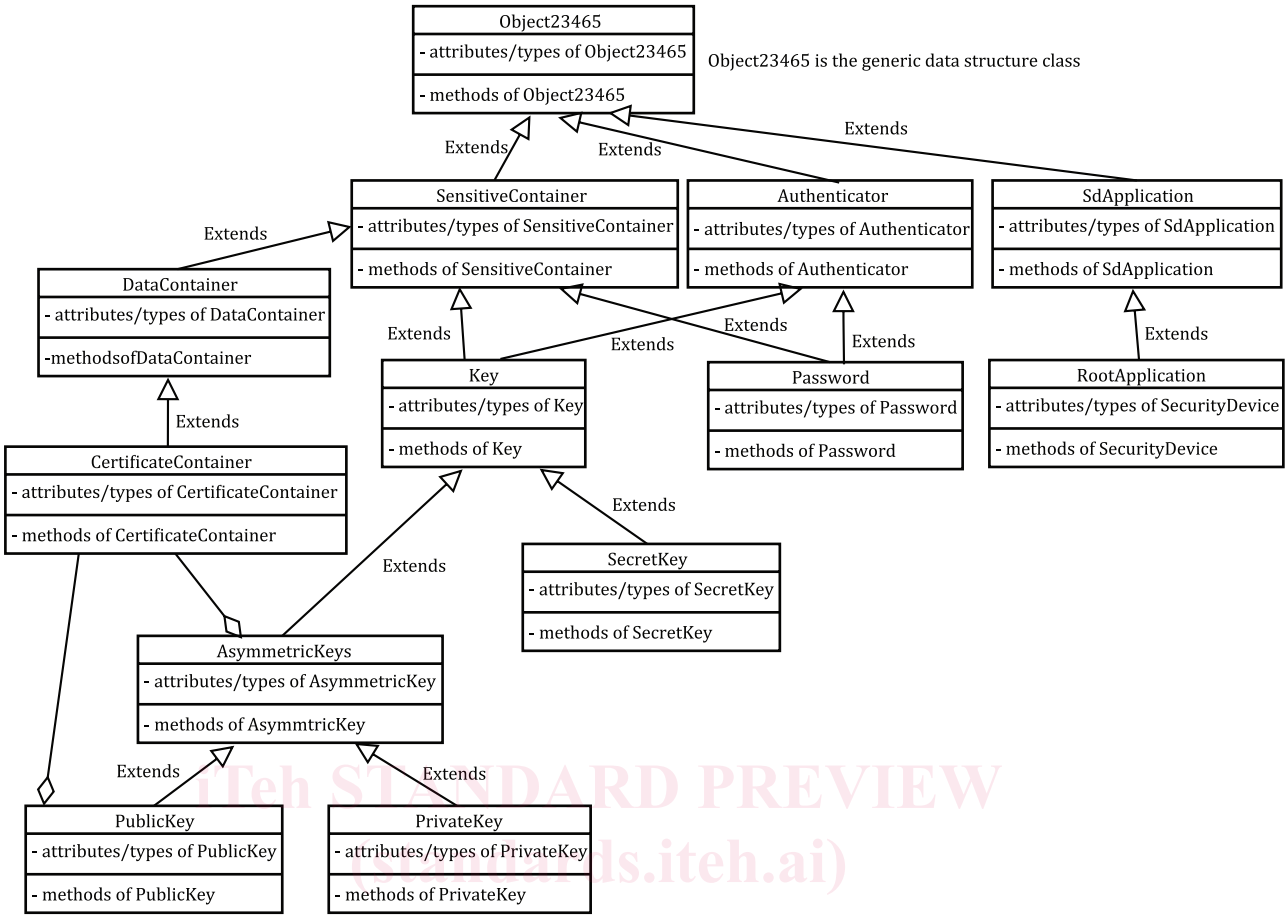

Figure 2 — General class diagram

9.2 Attributes and types

This document does not specify the implementation of classes from Figure 2. In general, neither class attributes nor instance attributes are specified in this document. The API defines getters and setters for handling relevant information.

9.3 Methods

The methods of each class are outlined in subclause 10.4.

9.4 References/instances

Addressing objects of the security device is achieved by using references to the instances of classes depicted in Figure 2. Programming languages often uses handles as the references to resources. Instead of these handles, references to class instances are used in the ISO/IEC 23465 series of standards.

ISO/IEC TS 23465-3[1)] describes the mechanisms of instantiation of the physical objects in the referenced security device.

1) Under preparation. Stage at the time of publication: ISO/IEC DTS 23465-3.