Administrator Documentation

CardOS API

V5.1 for Linux

Release Notes

Edition 03/2012

Atos IT Solutions and Services GmbH
Otto-Hahn-Ring 6
D-81739 Munich
Germany

Contact:
Atos IT Solutions and Services GmbH
Smartcard Solutions
Otto-Hahn-Ring 6
D-81739 Munich

Germany

http://www.atos.net/cardos

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual is reviewed regularly and any necessary corrections are included in subsequent editions. Suggestions for improvement are welcome.

Some of the specifications described herein may not be currently available in all countries. Please contact your Atos IT Solutions and Services GmbH sales representative for the most current information.

Contents

1 Introduction

This document provides system requirements, as well as known issues, for installation and operation of CardOS API for Linux. It is recommended to review this information before installing.

1.1 Target Group

The CardOS API - Release Notes are intended for system integrators and system administrators.

1.2 Third-Party Copyrights

The Cryptographic Token Interface Standard “RSA Security Inc. Public Key Cryptography Standard (PKCS)” is copyright by RSA Security Inc. (www.rsasecurity.com).

CardOS API uses decompression routines from the *lab* project (http://www.gzip.org/zlib). The zlib software is copyright by Jean-loup Gailly and Mark Adler.

CardOS API uses GNU Multiple Precision Arithmetic Library for bignum operations (http://gmplib.org/). The GMP library is copyright by Free Software Foundation, Inc., and licensed under LGPL v2.1 and v3.0

Please refer to the Open Source Software information document distributed on the product CD for more information on Open Source Software components used by CardOS API.

2 Subject of this Release

2.1 Released Version

These release notes are for:
CardOS API V5.1 for Linux (Build 22) for Linux
Released in March 2012
CD-Name: CardOSAPIV5.1L22

2.2 Licensed Software

CardOS API is a licensed product. The number of installed clients is limited to the number of licenses you own. The terms and conditions of the end user license agreement are displayed, while software is installed.

2.3 Supported Cryptographic Interfaces

CardOS API V5.1 for Linux supports the PKCS#11 Cryptographic Token Interface Standard v2.11.

2.3.1 PKCS#11 Cryptographic Token Interface Standard v2.11

CardOS API V5.1 for Linux supports the PKCS#11 Cryptographic Token Interface Standard v2.11. The PKCS#11 specification is available from the RSA web server at: ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v211/pkcs-11v2-11r1.pdf.

The CardOS API implementation covers all mechanisms, algorithms, and functions as defined in section 5, 6, and 8 of the "PKCS#11: Conformance Profile Specification" (ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/pkcs11Conformance.pdf).

CardOS API V5.1 for Linux supports the object types:

Object Type	Description	On Card Storage
CKO_CERTIFICATE	X.509 certificates[1]	yes
CKO_DATA	Plain data objects	yes
CKO_PRIVATE_KEY	RSA private keys	yes
CKO_PUBLIC_KEY	RSA public keys[2]. Calculations using these types of objects are always done in software.	yes
CKO_SECRET_KEY	Secret session keys	no

CardOS API V5.1 for Linux supports the following cryptographic mechanisms:

Mechanism	Mechanism Strength	On Card / Software Computation
CKM_RSA_PKCS_KEY_PAIR_GEN	256 … 2048bit[3]	on card
CKM_RSA_PKCS	256 ... 2048bit	on card
CKM_RSA_X_509	256 ... 2048bit	on card
CKM_MD5_RSA_PKCS	256 ... 2048bit	on card signing, software digest
CKM_SHA1_RSA_PKCS	256 ... 2048bit	on card signing, software digest
CKM_SHA224_RSA_PKCS	256 ... 2048bit	on card signing, software digest

[1] CKA_SUBJECT and CKA_ISSUER are read-only
[2] CKA_SUBJECT is read-only
[3] 1024bit maximum for CardOS/M4.01a

Mechanism	Mechanism Strength	On Card / Software Computation
CKM_SHA256_RSA_PKCS	256 … 2048bit	on card signing, software digest
CKM_SHA384_RSA_PKCS	256 … 2048bit	on card signing, software digest
CKM_SHA512_RSA_PKCS	256 … 2048bit	on card signing, software digest
CKM_MD5	128bit	software
CKM_SHA1	160bit	software
CKM_SHA224	224bit	software
CKM_SHA256	256bit	software
CKM_SHA384	384bit	software
CKM_SHA512	512bit	software
CKM_DES_KEY_GEN	56bit	software
CKM_DES_CBC	56bit	software
CKM_DES3_KEY_GEN	168bit	software
CKM_DES3_CBC	168bit	software
CKM_RC2_KEY_GEN	40 … 128bit	software
CKM_RC2_CBC	40 … 128bit	software
CKM_RC4_KEY_GEN	8 … 2048bit	software
CKM_RC4	8 … 2048bit	software
CKM_AES_CBC		software

CardOS API does not support functions to get and set the operation state of cryptographic functions (C_GetOperationState, C_SetOperationState). Dual-function cryptographic functions (C_DigestEncryptUpdate, C_DecryptDigestUpdate, C_SignEncryptUpdate, and C_DecryptVerifyUpdate) are not supported and need to be split up into separate calls. C_GetFunctionStatus and C_CancelFunction have been deprecated and will return CKR_FUNCTION_NOT_PARALLEL.

2.4 Card File System

2.4.1 CardOS API PKCS#15 Profile

The default card file system used by CardOS API V5.1 for Linux is based on the PKCS#15 Cryptographic Token Interface Standard v1.1. The PKCS#15 specification is available from the RSA web server at: ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-15/pkcs-15v1_1.pdf.

The file system is designed to allow the storage of 7 to 10 key pairs including the respective X.509 certificates. The actual amount of objects depends on object size (certificate size, key length) as well as the size of the corresponding object meta-data (like object names, object identifiers, etc.). The amount of meta-data per object is limited to 240 bytes. It is possible to customize the default memory allocation to customer specific needs.

For more details about the CardOS API file system please refer to the CardOS API - Developer Manual. The CardOS API - Developer Manual is available on request. Please contact your local sales agent.

2.4.1.1 Issuer PIN

Since HiPath SIcurity Card API 3.0 the file system is protected with a default Issuer PIN[4] that should be changed for custom projects.

Starting from HiPath SIcurity Card API 3.1 this Issuer PIN is initialized with the value passed for the SO PIN (also known as PUK) to `C_InitToken()`. This function is also used by Card Viewer to initialize cards.

The knowledge of the Issuer PIN is important in case you want to change the file system at a later stage.

[4] To avoid confusions with the so-called Admin Key of the Microsoft Smart Card CSP, the Admin PIN used in earlier manuals has been renamed to Issuer PIN.

In case the card is initialized using CardOS API V5.0 Windows Microsoft Minidriver interface the Issuer PIN is blocked after and cannot be used to modify the card after initialization.

2.4.1.2 PKCS#11 SO-PIN (PUK)

The SO-PIN (also known as PUK) is initialized with the value passed to `C_InitToken()`. Starting from HiPath SIcurity Card API 3.1 this value is also used to initialize the Issuer PIN (2.4.1.1).

It is possible to set the SO-PIN to a different value than the Issuer PIN by calling `C_SetPIN()` to assign a new value to the SO-PIN. This function is e.g. accessible from the Card Viewer function Change PUK. This does not affect the value of the Issuer PIN, which remains to the value originally assigned by `C_InitToken()`.

2.4.1.3 PKCS#11 SO-PIN (PUK) and Windows Base CSP Admin Key

Windows Base CSP introduces a Admin Key which takes the role of the PKCS#11 SO PIN (also known as PUK) for cards used with Microsoft Base CSP.

Since cards can either be initialized using the Microsoft Smart Card Base Cryptographic Provider or PKCS#11 it is required to set the authentication object which is not recognized by the respective interface to a defined value:

- For cards initialized with the Windows Base CSP (e.g. using Microsoft Certificate Lifecycle Manager) the SO PIN is not initialized and cannot be used.
- For cards initialized with PKCS#11 `C_InitToken()` the Windows Base CSP Admin Key is initialized to a value derived from the SO PIN passed to `C_InitToken()`. CardOS API uses the password based encryption scheme described in PKCS#12 appendix B[5].

  Example:

  SO-PIN: "1234567890"
  Admin-Key: `94346175 163e3d19 a4dfc1ae d55ee36b 942f3d15 3df46794`

2.4.2 GDOv1 Legacy File System

Cards initialized with the legacy GDOv1 file system used by CardOS API versions prior to 3.0 are supported in read-only mode by CardOS API V5.1 for Linux.

2.4.3 Carta Nazionale dei Servizi

CardOS API V5.1 for Linux supports the following CNS card profiles listed in the table below.

CNS cards can optionally contain a digital signature application stored inside DF_DS. The digital signature application needs to conform to the respective card certification description. CardOS API supports using the digital signature credentials stored inside the first DF_DS child application DF_DS_1.

Card Type	Basic Filesystem	Signature Application
CNS	Carta Nazionale dei Servizi, specified in “Carta Nazionale dei Servizi CNS; File System; Terza emissione 11 marzo 2005” issued by Centro Nazionale per l’Informatica nella Pubblica Amministratzione (CNIPA).	• CardOS DI V4.2B CNS Administrator Guidance • CardOS DI V4.2C CNS Administrator Guidance

[5] One iteration; Salt is set to the empty string.

Card Type	Basic Filesystem	Signature Application
PDC	Carta Nazionale dei Servizi, specified in "Carta Nazionale dei Servizi CNS; File System; Terza emissione 11 marzo 2005" issued by Centro Nazionale per l'Informatica nella Pubblica Amministratzione (CNIPA).	not applicable
HPC	HPC Lombardia, specified in "Carta Regionale dei Servizi SISS, Secondo Stadio, Specifiche del Servizio" Draft version, issued by Lombardia Integrata S.p.a; distributed October 2010	• CardOS V4.2 CNS Administrator Guidance • CardOS V4.2B CNS Administrator Guidance • CardOS V4.4 CNS Administrator Guidance

The following functionality is supported:

- Read-access to X.509 certificates stored on the CNS card
- Computation of RSA cryptographic operations
- Changing the user PIN (BSO_PIN_USR)
- Unblocking the user PIN using the user PUK (BSO_PUK_USR)
- Changing the digital signature PIN (PIN_DS)
- Unblocking the digital signature PIN using the digital signature PUK (PUK_DS)

Functions addressing the digital signature application are only available for cards with an activated digital signature application installed.

To use the digital signature application it is required to configure the secure messaging keys MASTER_KEY_SM_DS_KA und MASTER_KEY_SM_DS_KC.

These 3DES keys are specified in /etc/cardos_api.conf using following parameters:

```
CNSMasterKeySM_DS_KA_<iiii>_<gg>_<xx>
CNSMasterKeySM_DS_KC_<iiii>_<gg>_<xx>
```

where:

<iiii> is a 4 digit card group identifier set to the first 4 bytes of the card identifier stored in EF_ID_Carta
<gg> is a 2 digit card generation identifier. Shall be set to 00.
<xx> is a 2 digit identifier to DF_DS_x. Shall be set to 01.

Example:

```
CNSMasterKeySM_DS_KC_6030_00_01=4664c2291c7c029bf8763b1c345b4c344664c2291c7c029b
CNSMasterKeySM_DS_KA_6030_00_01=e0f7e3c7d0a11f3209511a2ad56292dae0f7e3c7d0a11f32
```

2.5 Released Objects

The tables below provide an overview of the documents, card initialization scripts, and binaries for Linux contained in a CardOS API distribution (n/a = not applicable).

Documents	Edition	Description
CardOS API - Release Notes - Linux.pdf	03/2012	This document describes system requirements for CardOS API Linux. As well, it provides information about supported applications, new features, and known issues.
CardOS API - Installation Manual	n/a	Describes the installation of CardOS API on Linux. This manual is provided as part of the HTML documentation contained on the product CD.
CardOS API - PIN Utility User Manual	n/a	Describes the use of the CardOS API PIN management utilities. This manual is provided as part of the HTML documentation contained on the product CD.

Card Initialization Scripts	Version	Description
InitTokenC804.cpd	5.1.13	Initialization script for CardOS/M4.01a cards
InitTokenC804.sig	5.1.13	Signature file for InitTokenC804.cpd
InitTokenC808.cpd	5.1.13	Initialization script for CardOS V4.3B cards
InitTokenC808.sig	5.1.13	Signature file for InitTokenC808.cpd
InitTokenC809.cpd	5.1.13	Initialization script for CardOS V4.2B cards
InitTokenC809.sig	5.1.13	Signature file for InitTokenC809.cpd
InitTokenC80A.cpd	5.1.13	Initialization script for CardOS DI V4.2B cards
InitTokenC80A.sig	5.1.13	Signature file for InitTokenC80A.cpd
InitTokenC80B.cpd	5.1.13	Initialization script for CardOS V4.2C cards
InitTokenC80B.sig	5.1.13	Signature file for InitTokenC80B.cpd
InitTokenC80C.cpd	5.1.13	Initialization script for CardOS DI V4.2C cards
InitTokenC80C.sig	5.1.13	Signature file for InitTokenC80C.cpd
InitTokenC80D.cpd	5.1.13	Initialization script for CardOS V4.4 cards
InitTokenC80D.sig	5.1.13	Signature file for InitTokenC80D.cpd

Linux x86/amd64 Platform Binaries	Version	Description
libcardossc.so	n/a	Card access library
libcardosui.so	n/a	Common smart card dialogs library
libcardos11.so	1.14	PKCS#11 library for CardOS smart cards
libcardos15.so	n/a	PKCS#15 library
libcardosxc.so	n/a	CNS file system support library
libcardosxg.so	n/a	GDOv1 file system support library
libgmp.so.3.5.2[6]	3.5.2	GNU Multiple Precision Arithmetic Library
libcardoscl.so.1.1.5	1.1.5	CES cryptographic library
cardospn	5.1.13	PIN management utility

[6] The GMP library is copyright by Free Software Foundation, Inc., and licensed under LGPL v2.1 and v3.0. The GMP source code and detailed licensing information is contained on the CardOS API distribution media.

3 System Requirements

This section lists the hard- and software requirements that are a prerequisite to run CardOS API V5.1 for Linux. Please ensure that the following hard- and software prerequisites are fulfilled before installing CardOS API.

3.1 Supported Smart Cards

CardOS API V5.1 for Linux supports the following smart cards from the CardOS family:

Smart Card OS	ATR	Required Packages
CardOS cards to be used with CardOS API PKCS#15 Profile (cf. 2.4.1)		
CardOS/M4.01a[7]	3b f2 98 00 ff c1 10 31 fe 55 c8 04 <TCK>	Latest service package
CardOS V4.3 B	3b f2 18 00 02 c1 0a 31 fe 58 c8 08 <TCK>	Latest service package, Verify RC package[8]
CardOS V4.2 B	3b f2 18 00 02 c1 0a 31 fe 58 c8 09 <TCK>	Latest service package
CardOS DI V4.2 B[9]	3b f2 18 00 02 c1 0a 31 fe 58 c8 0a <TCK>	Latest service package
CardOS V4.2 C	3b f2 18 00 02 c1 0a 31 fe 58 c8 0b <TCK>	Latest service package
CardOS DI V4.2 C[10]	3b d2 18 02 c1 0a 31 fe 58 c8 0c <TCK>	Latest service package
	3b 88 80 01 00 00 c8 0c 77 83 a1 00 <TCK>[11]	
CardOS V4.4	3b d2 18 02 c1 0a 31 fe 58 c8 0d <TCK>	Verify RC package, FRN package
CardOS cards to be used with GDOv1 Legacy File System (cf. 2.4.2)		
CardOS/M4.01a	3b f2 98 00 ff c1 10 31 fe 55 c8 04 <TCK>	Service package
Cards to be used with Carta Nazionale dei Servizi (cf. 2.4.3)		
CardOS V4.2 HPC	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 12 01 48 50 43 00 31 80 <TCK>	Refer to the respective administrator guidance or digital signature application certification documentation.
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 12 02 48 50 43 00 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 02 01 48 50 43 00 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 02 0a 48 50 43 00 31 80 <TCK>	
CardOS V4.2 B HPC	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 12 02 48 50 43 00 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 02 0a 48 50 43 00 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 09 01 12 02 48 50 43 00 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 09 01 02 0a 48 50 43 00 31 80 <TCK>	
CardOS V4.2	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 11 01 43 4e 53 10 31 80 <TCK>	

[7] RSA up to 1024bit
[8] The package can be installed optionally. It is not shipped as part of the CardOS API default initialization script.
[9] Released for contact based card operation with CardOS API only.
[10] Including CardOS DI V4.2 C CL
[11] ATQB used for contact-less operation.

PDC	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 05 01 01 01 43 4e 53 10 31 80 <TCK>	
CardOS V4.2 B PDC	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 09 01 11 01 43 4e 53 10 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 09 01 01 01 43 4e 53 10 31 80 <TCK>	
CardOS DI V4.2 B PDC	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 0a 01 11 01 43 4e 53 10 31 80 <TCK>	
	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 0a 01 01 01 43 4e 53 10 31 80 <TCK>	
	3b 88 80 01 20 43 4e 53 33 81 b1 00 <TCK>	
CardOS DI V4.2 B CNS	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 0a 01 21 01 43 4e 53 10 31 80 <TCK>	
	3b 88 80 01 20 43 4e 53 33 81 b1 00 <TCK>	
CardOS DI V4.2 C CNS	3b ff 18 00 ff c1 0a 31 fe 55 00 6b 05 08 c8 0c 01 21 01 43 4e 53 10 31 80 <TCK>	
	3b 88 80 01 20 43 4e 53 77 83 a1 00 <TCK>	
CardOS V4.4 HPC	3b df 18 02 c1 0a 31 fe 58 00 6b 05 08 c8 0d 01 12 02 48 50 43 00 31 80 <TCK>	
	3b df 18 02 c1 0a 31 fe 58 00 6b 05 08 c8 0d 01 02 0a 48 50 43 00 31 80 <TCK>	
ST/Incard PDC	3b ff 18 00 ff 81 31 fe 55 00 6b 02 09 02 00 01 01 01 43 4e 53 10 31 80 <TCK>	
	3b ff 18 00 ff 81 31 fe 55 00 6b 02 09 02 00 01 11 01 43 4e 53 10 31 80 <TCK>	
	3b ff 18 00 ff 81 31 fe 55 00 6b 02 09 02 00 01 01 01 43 4e 53 11 31 80 <TCK>	
	3b ff 18 00 ff 81 31 fe 55 00 6b 02 09 02 00 01 11 01 43 4e 53 11 31 80 <TCK>	
Athena PDC	3b df 18 00 81 31 fe 7d 00 6b 15 0c 01 80 01 01 01 43 4e 53 10 31 80 <TCK>	
	3b df 18 00 81 31 fe 7d 00 6b 15 0c 01 80 01 11 01 43 4e 53 10 31 80 <TCK>	

Please refer to the corresponding CardOS Packages & Release Notes for the latest appropriate service packages and their versions.

3.2 Linux

Assure that your computer meets the minimum conditions of the used Linux distribution and has at least 20 MB free disk space before starting the installation.

3.2.1 Platform Prerequisites

CardOS API V5.1 for Linux Linux runs on 32-bit x86 and 64-bit amd64 versions of Linux. CardOS API has been designed and tested for the compatibility to the minimum interface versions listed in the table below. Any later versions of these interfaces may work as far as they provide backward compatibility to the listed versions.

CardOS API Binaries	System Prerequisites
CardOS API V5.1.22 Linux x86	Kernel 2.6.26, glibc 2.7.18, PCSC-Lite 1.5.5[12], X.org 7.3[13]
CardOS API V5.1.22 Linux amd64	Kernel 2.6.26, glibc 2.7.18, PCSC-Lite 1.5.5, X.org 7.3

[12] PCSC-Lite should not be run in parallel with the OpenCT daemon.
[13] CardOS API uses lazy linking for X11 libraries. This allows using most of the CardOS API functionality even on systems where X11 is not available. Functions depending on UI (e.g. entry of a secondary authentication PIN) will fail in case the X11 dependencies cannot be fulfilled at runtime.

3.2.2 Supported Smart Card Readers

CardOS API PKCS#11 for Linux requires PCSC-Lite. It is recommended to use a PCSC-Lite version >= 1.5.1.

In general all fully PC/SC compliant readers operate successfully that provide a driver for Linux based on the PCSC-Lite framework (see section 7 *Known Issues* on page 17). Please note that in order to use RSA 2048bit your reader driver needs to support extended APDUs.

3.2.3 Supported Applications

CardOS API works with all applications using the PKCS#11 Cryptographic Token Interface Standard v2.11. A limitation of the functionality can occur, if the application is not completely standard compliant or uses undocumented proprietary functions of other vendors.

4 Changes Compared to Previous Releases

CardOS API V5.1 for Linux is a successor of the CardOS API V5.0 B for Linux.

4.1 New Features

CardOS API V5.1 for Linux
- Support of PKCS#15 profile as well as CNS smart cards.

CardOS API V5.0 B for Linux
- Added support for CardOS V4.2C DI. CardOS V4.2C DI can be used in both contactless and contact based operation scenarios after release of CardOS DI V4.2 C.

CardOS API V5.0 for Linux
- Supports 32-bit as well as 64-bit amd64 Linux operating systems
- Added support for CardOS V4.4 smart card operating system
- Extended multilanguage user interface including English, German, French, Spanish, Italian, Portuguese, and Slovak.

4.2 Fixed Bugs and Implemented Change Requests

CardOS API V5.1 for Linux:
- Fixed changing user PIN for CNS smart cards (LWVMAH).
- Improved interoperability with SUN Java PKCS#11 wrapper (LXBPPL).
- Support variable PIN length for dedicated CNS profiles (LZYQ7Z).
- Added legacy functionality for CNS project (LBRLRJ).

CardOS API V5.0 B for Linux:
- Fixed computation of combined PKCS#11 digest/signature mechanisms (e.g. `CKM_SHA1_RSA_PKCS`) (JXPPXB).
- Allow specification of attribute `CKA_OBJECT_ID` with `C_CreateObject(CKO_DATA)` (KBMLVG).
- Fixed memory violation in PKCS#11 logging (JNSM3J).
- Fixed synchronization of `C_GetTokenInfo()` with concurrent PKCS#11 calls (K1MOVL).

4.3 Migration from Previously Released Versions

CardOS API V5.1 for Linux is a maintenance release for CardOS API V5.0 for Linux. Please note the following items when migrating to the current version.

4.3.1 Update

If any previous release is installed on your system, you have to remove this version before installing CardOS API V5.1 for Linux.

4.3.2 Card File System

CardOS API V5.1 for Linux (as CardOS API V3.x) uses a PKCS#15-based file system by default (see section 2.4.1).

- All versions of CardOS API above V3.0 support the full functionality of previous V3.x releases with cards initialized using these previous versions respectively. However, features introduced with later releases than the release, which was used to initialize the card, may not be fully supported (please refer to section 4.1 *New Features* on page 13 for more details).
- CardOS API V5.1 for Linux supports all CardOS smart cards that have been initialized by previous Card API V3.x versions. Card API <= V3.2 might not support smart cards initialized by CardOS API V5.1 for Linux.

For CardOS cards initialized with CardOS middleware prior to HiPath SIcurity Card API V3.0 there is only a limited compatibility:

- Using the CardOS API V5.0 PKCS#11 interface GDOv1 cards can still be used as read-only.
- Cards initialized with the GDOv2 file system (introduced with Card API V2.2.0) are not supported by CardOS API V3.0 and above.

5 Ensuring Interoperability between PKCS#11 and Microsoft CSP

CardOS API has been designed to access the same smart card and objects stored on the smart card with both PKCS#11 and Microsoft CSP applications. Since the approach for the PKCS#11 architecture is quite different from that used by the Microsoft CSP architecture the points listed below should be considered to ensure proper interoperability.

The following items apply independent of the actual CardOS API target operating system platform. Especially you should consider the following in case you introduce your smart cards in a PKCS#11 environment and plan to use the same smart cards with Microsoft CSP applications in the future.

Please contact your system integrator for further details.

5.1 PKCS#11 Object Identification and Microsoft CSP Container Names

- In PKCS#11 the attributes CKA_ID and CKA_SUBJECT are used to uniquely identify an object. This allows a PKCS#11 application to e.g. identify keys and corresponding certificates.
- Microsoft CSP uses so called containers to store keys and corresponding certificates. The containers are uniquely identified by container names.
- The PKCS#11 applications (e.g. Firefox) usually display the value of CKA_LABEL for key and certificate objects. The CSP applications (e.g. Internet Explorer) usually display the container name for the key and certificate objects. CardOS API maps the container name from the PKCS#11 attribute CKA_LABEL. This allows users to uniquely identify keys and certificates independent of the used application.

The chosen approach results in the following consequences:

- Each container can only hold a single asymmetric key pair and a corresponding certificate. This does not impose any limitation on your PKI design, since all known CAs (and CA products) are able to create a new container for each key pair.
- PKCS#11 objects (a Certificate, a Public and a Private Key object) with an identical CKA_LABEL are pooled to one container. The containers itself are distinguished using different CKA_LABELs. In case two objects of the same type (e.g. two certificates) share the same CKA_LABEL it is not possible to use these PKCS#11 objects via the Microsoft CSP interface.

  This affects you if on one hand the PKI enrolls the key material via the PKCS#11 interface and, on the other, the applications access the key material via Microsoft CSP.

5.2 RSA Public Exponent Length

The data structure used by the Microsoft CSP framework to represent RSA keys does not allow the public exponent to exceed the length of 32 bits. In case you plan to generate RSA key pairs via the PKCS#11 interface and these keys should be accessible via the Microsoft CSP interface the public exponent passed as CKA_PUBLIC_EXPONENT to `C_GenerateKeyPair()` should therefore not exceed the length of 32 bits (4 bytes).

5.3 Unblock User PIN

Depending on the initialization of the card the User PIN can either be unblocked using the Admin Key via the Microsoft Minidriver interface (e.g. by Microsoft Identity Lifecycle Manager) or by using the SO-PIN (PUK) via PKCS#11 (e.g. CardOS API - Viewer, PKCS#11 PIN Tool). Refer to 2.4.1.3 for details.

6 Security Considerations

Since CardOS API is likely to be used in environments that deal with sensitive data the following items should be considered before using CardOS API:

- The MF of the file system which is created by CardOS API (via the PKCS#11-function `C_InitToken()` that is used as well by the CardOS API - Viewer to initialize cards) is protected by an Issuer PIN (see section: 2.4.1.1 *Issuer PIN* on page 6).
  The knowledge of the Issuer PIN allows installing additional packages on the CardOS smart card. These packages may contain malicious code that can be used to break the smart cards security.
  It is strongly recommended to review and if deemed necessary adjust the access conditions of the cards MF before using CardOS API in a productive environment.

- CardOS API uses the standard PC/SC stack of your workstation to communicate with the smart card. This communication might be intercepted to spy out sensitive information (e.g. PINs, results from decryption operations) or to pretend wrong results (e.g. successful verification of an invalid signature).
  It is strongly recommended to take effective measures to protect the overall system security to keep Trojan Horses and other malicious software out of your system.

- In case you intend to use CardOS API in networking scenarios (Remote Desktop, Terminal Server) proper action need to be taken to secure your network communication.

7 Known Issues

The following items describe known limitations and interoperability issues of the current version of CardOS API. The list includes all known issues independent of the actual CardOS API target operating system platform. It has been numbered for easier reference. The assigned numbers do not reflect any severity or priority assigned to the issue.

1. The PKCS#11 Cryptographic Token Interface Standard does not support hot-plugging of card readers or USB-Tokens. All applications using the CardOS API must be closed and restarted, if a new reader has been installed or has been removed from the system.

2. Some smart card readers are not able to access CardOS smart cards with the highest baud rate suggested by the CardOS ATR. The affected readers do not negotiate a connection at a slower baud rate and communication with the smart card does not work. The CardOS ATR can be configured to suggest a fixed baud rate (e.g. 9600 bit/s and up) to avoid those problems at the cost of system performance. Refer to the CardOS User's Manual for more information.

3. The IFD handlers of some smart card readers do not implement proper support for the ISO time wait (WTX) command. These readers fail if executing time consuming commands such as `GENERATE KEY PAIR`.

4. Terminal service roaming users between systems with different peripherals (card readers) may require restarting the applications using CardOS API after reconnecting to the server.

5. Depending on the Citrix ICA protocol version used the Citrix ICA protocol does not support the use of more than one USB reader on a Citrix ICA Client.

6. It may be required to restart CardOS API automated certificate propagation after reconnecting a Citrix remote session.

7. The *Initialize Card* function (`C_InitToken()`) may not terminate correctly on Mac OS X, Linux, or when running in a Citrix remote session.

8. The use of Secondary Authentication PINs is currently not supported by Mac OS X Tokend.

9. Private keys protected by a Secondary Authentication PIN should neither be used for Windows Smart Card Logon, the function *Run as…*, nor for EFS (Encrypted File System).

10. The Microsoft Vista functions *Run as…* and EFS (Encrypted File System) do not work properly in case more than one smart card is inserted.

11. The Preview Pane of Outlook Express should be deactivated.

12. The Preview Pane of Windows Mail for Windows Vista should be deactivated.

13. The Unblock Sec Auth PIN function cannot be supported for Cherry ST-2000 reader with the firmware releases lower than 05.11. In order to use this functionality, the firmware of the reader must be updated to release 05.11 or later.

14. Details of PIN pad readers supporting the PC/SC interface:
    a. If a timeout occurs while entering a PIN, it might be treated as a Cancel.
    b. Depending on the reader you use, the displayed windows and the respective workflow might be different, but the number and type of the PIN entries are the same.
    c. Depending on the reader you use, the PIN entries, exceeding the configured max. PIN length, are treated as wrong input or are truncated to the maximum length without a warning message.
    d. Not all readers support timer configurations.
15. The readers Omnikey CardMan 3621 and the Fujitsu Keyboard Reader KBPC CX D do not work properly with Vista.
16. On Windows Vista some scenarios do not allow to grant sufficient access privileges to CardOS API to create log files.
17. The installer includes a package with the required Microsoft Visual C++ 2005 runtime libraries. In case of these libraries are missing after the installation, the Visual C++ 2005 Redistributable Package must be reinstalled. The libraries for x86 platforms are available from Microsoft as `vcredist_x86.exe` (http://www.microsoft.com/downloads/details.aspx?familyid=32bc1bee-a3f9-4c13-9c99-220b62a191ee&displaylang=en). The libraries for x64 platforms are available from Microsoft as `vcredist_x64.exe` (http://www.microsoft.com/downloads/details.aspx?familyid=90548130-4468-4BBC-9673-D6ACABD5D13B&displaylang=en).
18. The default CCID IFD handler shipped with Microsoft Windows does not properly support extended APDUs. This means that 2048bit keys cannot be used. In case you are affected by this limitation you should install the IFD handler provided by your card reader vendor.
19. CardOS API Installer sets its display language depending on the *Language for non-Unicode programs*. A setting for *Regional and Language Options*. This may result in a display language different from your native Windows system.
20. Keys generated with the "Strong Key Protection" flag using the CardOS API V5 Minidriver are not protected with a special Secondary Authentication PIN. Microsoft Base CSP does not provide the required information to the Minidriver (`CRYPT_USER_PROTECTED`).
21. Some PKCS#11 Applications (e.g. Firefox) are not able to handle multiple cards with the same token label. Keep that in mind when initializing your cards.
22. Certificate size shall not exceed 4096 bytes.
23. When creating new objects (keys, certificates) from concurrent processes the other process may only recognize these changes after a re-insertion of the smart card.
24. In some scenarios (e.g. Adobe Reader) the Microsoft Base Smart Card Crypto Service Provider for Windows XP does not properly authenticate to the card before computing a signature. In this cases `CardSignData()` fails with `SCARD_W_SECURITY_VIOLATION` (`0x8010006`). The CardOS API Minidriver configuration setting `CMFlags = 8` may be used as a work-around.
25. CardOS API - Viewer is not able to handle more than one secondary authentication PIN. In case you want to manage more than one secondary authentication PIN use the CardOS API PKCS#11 PIN Utility.
26. Removing the smart card during modification of a PIN with CardOS API - Viewer using a secure PIN entry (SPE) device may result in undefined behavior of CardOS API - Viewer.
27. In case C_InitToken() is called to initialize a card using a secure PIN entry (SPE) device the new SO PIN (PUK) assigned to the token must have a length of 10 characters.

28. Thunderbird automatically tries to update opened encrypted mails as soon as a smart card is inserted. This results in asynchronous PIN prompts initiated by the Thunderbird process. This behavior can be moderated if the message pane is disabled and Thunderbird is configured to open new messages in tabs instead of separate windows.

29. When your system is put to sleep/hibernate mode all active PKCS#11 sessions are closed for security reasons. Operations running in these sessions terminate with an error.

30. If Acrobat Reader is used with PKCS#11, Acrobat Reader does not properly recognize the certificate stored on a card after removing and re-inserting the card. We recommend to use Acrobat Reader using the Microsoft CSP interface where possible.

Please also refer to the list of Known Issues contained on the installation media that may provide additional last minute information and change notes.

8 Reporting Bugs

Please use the accompanying Ticket Registration form (located at the CD-ROM in PDF and Word format) for a trouble report or change request. Send this filled out form to who is listed in the service contract (or partner contract). In case you have no service contract please contact your local Sales Agent.

You should have the following information to hand before you start filling in the Ticket Registration form:

- *Contact Information*
  Company, Country, Name, E-mail, Service Contract No., and customer reference ticket.
- *Maintenance Contract Number*
  Number identifying your maintenance contract with your Sales Representative or Atos IT Solutions and Services GmbH.
- *Description*
  Product Name, Version number, Build number, and description of the problem.
- *System Information*
  OS Platform, OS Version, Processor Type, Smart Card Type, and Smart Card Reader.
- *Environment Information*
  Description of the architecture (e.g. Client/Server, Workflow, etc.).
- *Attached Files*
  Card Log or the Local Log file as explained in the Installation Manual.

9 Glossary

The following terms and abbreviations are used in the documents of the CardOS API software distribution:

API	Application Programming Interface (API) is an interface that can be used by programs either to control hardware devices or functions of the operating system.
Base CSP	Smart Card Base Cryptographic Service Provider
CA	A certification authority, or CA, issues certificates and binds the identity of it to a person or computer.
CAPI	Microsoft Cryptographic API; also called Crypto API
CardOS	Operating System for smart cards being developed by Atos IT Solutions and Services.
Certificate	A digital certificate is a file that includes the name of the certificate holder, dates of validity, a Public Key, and the name of the certificate issuer.
Cryptoki	→ PKCS#11
CSP	Cryptographic Service Provider (CSP). A CSP is responsible for the creation of keys and their use for different tasks. Several different CSPs can be installed on one PC, which differ for example in key length, algorithms for encryption, or the inclusion of smart cards.
Data Object	A Data Object is a file that can be imported or exported from the smart card.
DF	A DF (dedicated file) is a directory in the file system of a Smart Card.
Digital Signature Application	A Digital Signature Application consists of appropriate file-structures and objects on a smart card, that enables the computation of digital signatures.
DIN NI 17-4	Specification of the interface to smart cards with Digital Signature Application compliant to SigG and SigV.
ICC	Integrated Circuit Card. ISO compliant description for a Smart Card.
ICCSP	An Integrated Circuit Card Service Provider (ICCSP) is responsible for the allocation of the functionality of a smart card, independent of the smart (ICC) card operating system.
MF	A MF (master file) is the root directory in the file system of a smart card.
Minidriver	Minidriver presents a consistent interface to smart cards to the Microsoft Smart Card Base Cryptographic Service Provider.
OID	An object identifier is a registered number sequence to uniquely identify an object type.
PC/SC	Interoperability Specification for ICCs and Personal Computer Systems.
PIN	The Personal Identification Number (PIN) is used to authenticate you as the owner of the card. Once a correct PIN is entered, the error counter is reset.
PIN pad	Especially on security relevant applications (e.g. money transactions) the entry of a PIN is subject to regulations for keyboards. Those specific keyboards are called PIN pad. They are mechanical and cryptographic protected, so the PIN can't be eavesdropped during entering. Smart card readers with a built-in PIN pad are called PIN pad readers.
PKCS#11	The Public-Key Cryptography Standards (PKCS) are specifications that were developed by RSA Security in conjunction with system developers worldwide. PKCS#11 defines a technology independent programming interface for cryptographic devices such as smart cards.
PSE	Personal Security Environment - Security relevant information is stored in a PSE. Among other things it contains the certificate and the private key of the card holder and it might contain one or several CA-certificates. The PSE can take the form of an encrypted file or a smart card and is protected by a password.

PUK	The Personal Unblocking Key (PUK), also known as a Super-PIN or SO PIN (see PKCS#11 standard), is used to change or to unblock the PIN.
Secondary Authentication PIN	The intent of secondary authentication is to provide a means for a smart card to produce digital signatures for non-repudiation with reasonable certainty that only the authorized user could have produced that signature. A Secondary Authentication PIN must be presented each time a particular signing key is used to perform its digital signature operation. Depending on the security requirements of the application a Secondary Authentication PIN has to be entered via e.g. a PIN Pad reader that bypasses the workstation.
SigG	Germany's Electronic Signature Act, which went into effect in May 2001, defines the framework conditions for electronic signatures.
SigV	Germany's Signature Ordinance. Supplement to the SigG with regard to the procedures of the certification authorities and products to be used for certificate issuance and signature creation; effective from November 2001.
SO PIN	Security Officer PIN. This definition is used in the PKCS#11 standard → PUK.
SPE	Secure PIN Entry (SPE) can be achieved by using of a PIN pad reader.
Super-PIN	→ PUK
Token	A token is an object containing the security information for a cryptographic session. A smart card can be such a cryptographic token.
Transport PIN	The Transport PIN is usually provided by a Trust Center over a trustworthy channel (e.g. a PIN letter). Before using a Digital Signature Application the owner of the smart card has to initialize a personal Digital Signature PIN to the card. For this initialization a so-called Transport PIN has to be entered to assign a personal Digital Signature PIN and Digital Signature PUK to the card.
VBS	Visual Basic Script
WinSCard API	Windows Smart Card client API