



**WIS@key Smart Security Interface Utility™**

**V4.8.1**

**Manual**

## Contents

1	Preface .....	4
2	About this Manual .....	5
2.1	Installation Requirements .....	5
2.2	Supported Smart Cards .....	6
2.3	Tested Card Readers .....	7
2.4	Secure Pin Entry (SPE) .....	7
2.5	Unattended Installation .....	8
3	User Tool: WIS@key Smart Security Interface Utility .....	9
3.1	Change PIN .....	9
3.2	Unlock PIN .....	10
3.3	Change Token SO PIN .....	10
3.4	Registration .....	11
4	Register Tool .....	12
4.1	Start WSSI Utility .....	12
4.2	PKCS11 register/ unregister .....	12
4.3	Pause/ Continue .....	13
4.4	Settings .....	13
4.5	About .....	14
4.6	Exit .....	14
5	WIS@key Extension Tool .....	15
6	CSP of WIS@key Smart Security Interface .....	16
6.1	General Proceedings .....	16
6.2	Smart Card Login to a Windows 2000 Domain .....	16
6.3	SSL- Authentication with Smart Card over the Internet Explorer .....	16
6.4	Outlook Express with Electronic Signature and Encryption via Smart Card .....	17
6.5	Windows VPN-Login with Smart Card .....	17
7	PKCS#11-Module of WIS@key Smart Security Interface .....	18
7.1	General Methodology .....	18
7.2	Smart Card Login to a Novell eDirectory (formerly NDS) .....	18
7.3	SSL- Authentication with Smart Card over Netscape .....	18
7.4	Email-Security by Smart Cards and Tokens with Netscape's Messenger .....	19
8	References .....	20
9	Information / Export Restrictions .....	21
Appendix A:	Reference for Developers .....	22
	Functions according to PKCS#11-Standard .....	22
	Synopsis of specific functions .....	23

C_Finalize.....	23
C_GetObjectSize.....	23
C_GetSlotList .....	23
C_GetTokenInfo .....	23
C_Initialize .....	24
C_InitToken .....	24
C_OpenSession .....	24
C_WaitForSlotEvent.....	24
Objects	25
Mechanism .....	27
Sign (RSA):.....	27
Verify (RSA):.....	27
Encrypt (RSA):.....	27
Decrypt (RSA): .....	27
Digest (Hashfunctions SHA1, MD2, MD5): .....	28
Appendix B: Non-Standard Functions in PKCS#11 DLL .....	29
Appendix C: Log Information.....	30
Convenience Files .....	30
Registry Settings .....	30

## 1 Preface

Thank you for purchasing the **WIS@key Smart Security Interface (WSSI)**.

**WIS@key Smart Security Interface Utility** provides modules that you need in order to integrate different smart cards and USB tokens into your applications. The following file structures (profiles) are supported:

- Charismathics corporate profile
- PKCS#15 profile
- Carta Nazionale dei Servizi (CNS) profile
- FINEID profile
- AET profile
- PIV profile

**WIS@key Smart Security Interface Utility** is comprised of the following modules:

- the User Tool - WIS@key Smart Security Interface Utility
- the Register Tool for automatic registration of certificates
- the WSSI Extension Tool for adding new ATR/OS associations
- the CSP
- the PKCS#11-module

The user tool **WIS@key Smart Security Interface Utility** allows changing your user PIN and registering your smart card or USB token.

**WIS@key Smart Security Interface-CSP** enables you to enhance applications and services in a Microsoft environment and their use with a smart card.

**WIS@key Smart Security Interface-PKCS#11** enables you to use additional applications and services that use this standard interface. PKCS#11-Modules are in use by Mozilla Firefox, Mozilla Thunderbird, and Netscape and in Novell environments.

In particular the following applications can be augmented by **WSSI**:

- smart card login to Windows Domains or Novell eDirectory
- SSL- Authentication by smart card (Internet Explorer, Mozilla Firefox, Netscape, ...)
- email security with cards (PGP, Netscape Messenger, Outlook, Outlook Express, Mozilla thunderbird ...)
- VPN with smart cards (Microsoft, Cisco, ...)

This manual is meant for system administrators. Application developers, who develop their own applications that access software modules of **WIS@key Smart Security Interface Utility**, e.g. PKCS#11, will find additional information in the appendices.

## 2 About this Manual

The **WIS@key Smart Security Interface Utility** is described in chapter 3: "User Tool: WIS@key Smart Security Interface Utility". It contains information on how to change PINs and register your smart card.

Furthermore, you will find additional information regarding the Register Tool, WSSI Extension Tool, CSP and PKSC#11 and which applications may be upgraded by hardware tokens. Application developers can find further information on how to access modules (e.g. accessing PKCS#11) of **WIS@key Smart Security Interface Utility** tool in the appendices, if they intend to develop a proprietary application. Certificate Attributes (Key Usage) is a concise description of the certificate attributes, i.e. information about key employment. However, an explanation on how to configure environments of Microsoft or other producers exceeds the scope of this manual. In these cases, please consult the documentation of the corresponding supplier.

**NOTE:** To understand this manual you need basic knowledge in IT-security. Especially, you should be familiar with the following notions: certificate, private and public key, secret key, digital signature, PKI, etc.

Before you can install **WIS@key Smart Security Interface Utility tool**, the card reader you intend to use must be installed according to the manufacturer's guidelines and be fully operational. The installation of **WIS@key Smart Security Interface Utility** is run from the program CD. Please execute the file SETUP.EXE as a user with administrator rights. Follow the installation instructions.

### 2.1 Installation Requirements

If not explicitly required otherwise in the following:

Microsoft Windows 2000 with Service Pack 4  
 or Windows XP with or without Service Pack 1, 2 and 3  
 or Windows Server 2003 with or without Service Pack  
 or Windows Vista with Service Pack with or without Service Pack 1, 2  
 or Windows Server 2008 with or without Service Pack

**Note:** During the installation process the CSP Module is registered automatically with the Windows operating system. If Netscape, Firefox or Thunderbird is installed on your computer the Register Tool can assist you in enabling PKCS#11 support within these applications. Refer to chapter 4: "Register Tool" for more detailed information.

The following applications are supported:

- Smart card login to a Windows 2000 or 2003 or 2008-Domain:  
 ADS, Enterprise CA, Windows 2000 or 2003 or 2008 Server and as Client: Windows 2000 Professional or Windows XP Professional
- SSL- Authentication with smart card using Internet Explorer:  
 Microsoft Internet Explorer 5.0, 5.5, 6.0, 7.0, or 8.0 High Encryption Pack, SSL V3 with Strong User Authentication
- Outlook with digital signature and encryption via smart card:  
 Outlook Express 5.0, 5.5 or 6.0,  
 Windows Mail  
 Outlook 2000, 2003, 2007

- Lotus Notes with digital signature and encryption via smart card:  
Lotus Notes 6.5 or higher
- Windows VPN-Login with smart card:  
Windows 2000 Server and as Client: Windows 2000 Professional or  
Windows 2003 Server and as Client: Windows 2000 or XP  
Windows 2008 Server and as Client: Windows XP or Vista
- Smart card login to Novell eDirectory:  
Netware 5.1 SP3, eDirectory 8.6.1, Novell Client 4.83 SP1, NMAS EE 2.0 (with the included Universal Smartcard Login Method) with NCI 1.5.7 (Server and Client), NMAS 2.1 (with the included Universal Smartcard Login Method) with NCI 2.4.1 (Server and Client) or higher in each case
- Smart card login to Lotus Notes:  
Lotus Notes 6.5 or higher
- SSL- Authentication with smart card with Netscape:  
Netscape Navigator 4.72 (High Encryption), 4.73, 4.76, 6.x
- Email-Security via smart cards with Netscape Messenger:  
Netscape Messenger 4.72 (High Encryption), 4.73, 4.76, 7.x  
Thunderbird 1.5 and above
- E-Mail-Security via PGP support (PKCS#11): PGP Personal Desktop 8.1 for Windows
- Compatibility/Smart card administration of the Baltimore-PKI (PKCS#11): Token Manager for Be-trusted Unicert V5.2 for Windows
- Compatibility/Smart card administration of the Entrust-PKI (PKCS#11): Security Manager Administration 7.0
- Compatibility/Smart card administration of the Ecos-PKI Appliance BB5000 (PKCS#11)

The mentioned products do not require any further client software. Please refer to the manual of your software application if it is not listed above.

Furthermore, WSSI ties in with the following preboot/ harddisk encryption environments:

CheckPoint/Pointsec	Mcafee	Secude
Ultimaco	PGP	

## 2.2 Supported Smart Cards

**WIS@key Smart Security Interface** supports the following smart cards/tokens:

ACOS A-Trust Card	ACOS EMV A03	ACOS SMARTMX
ActivIdentity Card	Axalto Cyberflex Access V2c	CardOS M4.01(a)
CardOS V4.20 / V4.2B / V4.2C / V4.4	CardOS V4.30 / V4.3B	G&D Sm@rtCafe Expett 64k
Gemalto EMV – PKI	GemXpresso Pro R3.2	JCOP 20, 21
JCOP 30,31	JCOP 41	jTOP JCX32/36
KONA	Micardo EC 2.x	NetKey E4/2000
NetKey PKS/2000/E4	Oberthur Cosmopo RSA V5.x	Oberthur CosmopolIC 64K V5.2
Oberthur Cosmo V5.2 PIV	Oberthur ID-one Cosmo V7.0	Oberthur Cosmo v5.4

plusID 60,75,90	Setec SetCard	Sm@rtCafe Expert 2.0, 2.1
Sm@rtCafe Expert 3.0/3.1, 3.2	StarCOS SPK 2.3, 2.4	StarCOS SPK 3.0
StarCOS 30	TCOS 2.x	CardLogix

### 2.3 Tested Card Readers

Please make sure your PC/SC smartcard reader has been installed according to the producer's specifications and is fully operational. **WIS@key Smart Security Interface Utility** has been tested with the following card readers:

ACS38 USB	charismathics plug'n'crypt	Eutron cryptoidentity CCID
Eutron Digipass 860	Fujitsu Siemens Computer Smartcase KB SCR PRO	Fujitsu Siemens Computer Smartcase KBPC CX
Fujitsu Siemens Computer Smartcase SCR USB	Fujitsu Siemens Computer Smartcase SCR USB internal	Fujitsu Siemens Computer Smartcase Token USB
KOBIL KAAAN advanced	Omnikey Cardman 1010 serial	Omnikey Cardman 2011 serial
Omnikey Cardman 2020 USB	Omnikey Cardman 3021 USB	Omnikey Cardman 3121 USB
Omnikey Cardman 3620 USB	Omnikey Cardman 3621/3821	Omnikey Cardmann CM4040 (PC-Card)
ORGA Card Mouse USB	SCM SCR 331 USB	SCM SCR 3310 USB
SCM SCR 3340 (Express-Card)	SCM SCR 532 serial/USB	SCM SCR241 PCMCIA
SCM SCR333	SCM SCR335 USB	

Additionally a great number of readers not explicitly mentioned above, but built upon compatible hardware, are supported.

Note:

- Only PC/SC-drivers are supported. There is no support for CT-API-drivers.
- If RSA 2048 bit key shall be used, then the smartcard reader must support the extended APDU.

### 2.4 Secure Pin Entry (SPE)

A number of card readers come equipped with their own PIN-pad. This PIN-pad can be used for SPE if one of the following devices is used:

Cherry Keyboard G83-6644	Omnikey 3621 USB	OmniKey 3821 USB
--------------------------	------------------	------------------

Please make sure your windows device drivers are up to date if you want to use PC/SC 2.0 with SPE.

To enable SPE, edit the registry setting at

[HKEY\_LOCAL\_MACHINE\SOFTWARE\WIS@key\smart security interface]

To activate SPE, use: "USE\_PINPAD"=hex:01

To deactivate SPE, use: "USE\_PINPAD"=hex:00

Although WSSI supports alphanumeric PINS in general, SPE obviously only supports digits. Please make sure the PINs used for the card can be entered by using SPE if you intend to use it.

## 2.5 Unattended Installation

Instead of calling setup.exe, the installation can also be started in unattended mode by calling the corresponding msi file from the setup directory

To install the user edition:

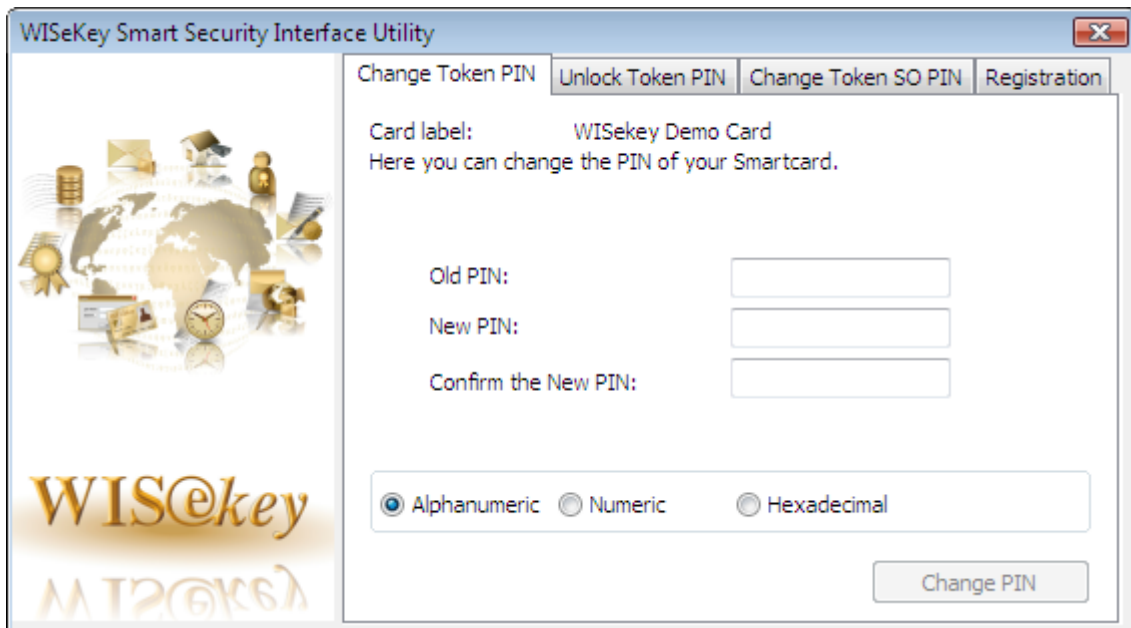
```
msiexec /i "WSSI x.x - user edition.msi" /qn
```

### 3 User Tool: WISEkey Smart Security Interface Utility

This tool exposes all relevant functions if you acquired **WISEkey Smart Security Interface Utility**. Changing your pin and the registration of your key/certificates of the smart card are available. Insert your smart card in the reader and open **WISEkey Smart Security Interface Utility** by following the path:

"Programs"->"WISEkey "->"smart security interface" ->"smart security interface utility".

#### 3.1 Change PIN



To change your PIN, insert the old PIN followed by the new PIN which must be entered a second time as confirmation. The minimum length of the User PIN is four characters and the maximal length is ten characters.

Click on the button "Change PIN", and you receive a window with the confirmation.

**IMPORTANT:** After three consecutive wrong inputs the User PIN will be locked. Please choose a PIN, which you can remember well, but which cannot be easily guessed. Avoid birthdays or simple sequences of numbers like 1234 or 1111.

### 3.2 Unlock PIN

WIS@key Smart Security Interface Utility

Change Token PIN | **Unlock Token PIN** | Change Token SO PIN | Registration

Card label: WIS@key Demo Card  
Here you can unlock the PIN of your Smartcard.

SO PIN:

New PIN:

Confirm the New PIN:

Alphanumeric  Numeric  Hexadecimal

Unlock PIN

To unlock your PIN, enter the SO PIN followed by the new PIN, which must be entered a second time as confirmation. The minimal length of the User PIN is four characters and the maximal length is ten characters.

Click on the button "Unlock PIN" and a confirmation window opens.

### 3.3 Change Token SO PIN

WIS@key Smart Security Interface Utility

Change Token PIN | Unlock Token PIN | **Change Token SO PIN** | Registration

Card label: WIS@key Demo Card  
Here you can change the SO PIN of your Smartcard.

SO PIN:

New SO PIN:

Confirm the New SO PIN:

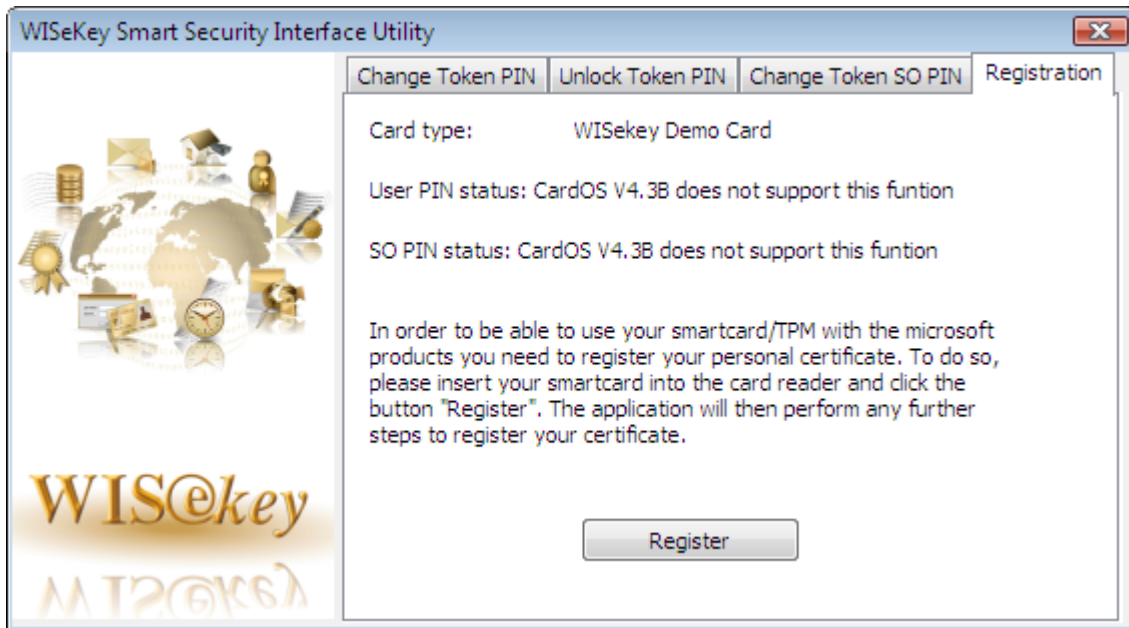
Alphanumeric  Numeric  Hexadecimal

Change SO PIN

To change the Token SO PIN, enter the SO PIN followed by the new SO PIN, which must be entered a second time as confirmation. The minimum and maximum length of the SO PIN is dependent on the card OS.

Click on the button "Change SO PIN" and a confirmation window opens.

### 3.4 Registration



Your smart card may contain multiple certificates and keys. These certificates must be registered once, so that applications can use these. Particularly if it concerns the registration of the certificate/keys with the Microsoft Windows certificate database.

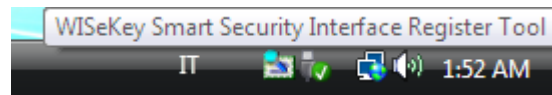
**IMPORTANT:** THE REGISTRATION NEEDS TO BE DONE ONLY ONCE FOR EACH CARD.

## 4 Register Tool

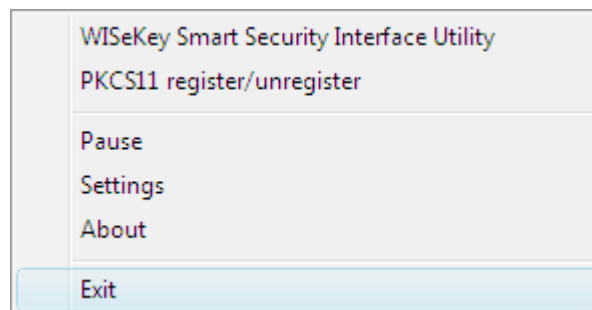
To make certificates accessible for Windows applications like Internet Explorer or Outlook Express, you can automatically register the certificates from your smart card in the certificate store of Windows. The settings for this registration are configured in this Register Tool.

The default functionality is as follows: as soon as a smart card is inserted into the card reader, the certificates are automatically registered, as long as the Register Tool is active. On smart card removal, the certificates are **not** automatically unregistered. If this is desired, you can adjust this using the "Settings".

You can call the Register Tool of **WIS@key Smart Security Interface** either over the Start menu or over the tray icon:



Then you get the possibilities of starting the WIS@key Smart Security Interface Manager or the WIS@key Smart Security Interface Utility to pause the Register Tool, to configure Settings, to read information or to terminate the Register Tool, which is now explained.



### 4.1 Start WSSI Utility

The context menu of the system tray icon offers the choice of starting the user edition (WIS@key Smart Security Interface Utility).

### 4.2 PKCS11 register/ unregister



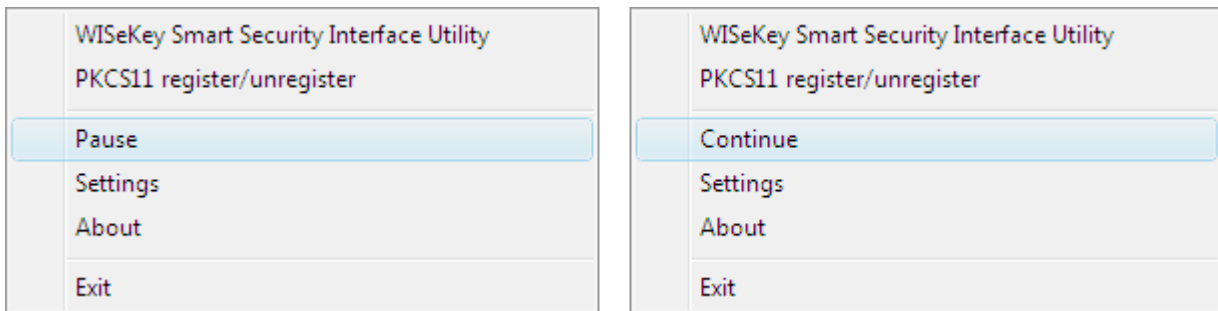
For the smartcard to be usable in the Netscape/Mozilla family of products a PKCS#11 module has to be registered with the products. This dialog offers a convenient way of installing the module. If the checkbox in front of the product name is not checked, the product is not configured to use the WISEkey PKCS#11 module. A marked checkbox signifies a PKCS#11 enabled product. Use the “register/unregister” button to apply the changes made.

A few things to keep in mind when using this feature:

- Firefox and Thunderbird share the same configuration files. Installing the module in Firefox enables it in Thunderbird as well.
- Most applications have to be closed at the time “register/unregister” is selected, otherwise the operation fails.
- It is possible that the Register Tool considers the PKCS#11 module registered, while it actually is not, or vice versa. This may be due to failed installation/un-installation attempts or manual changes to the configuration.  
In this case, repeat the “register/unregister” process (remember to close all confirmation windows), until the desired effect sets in. This should take no more than 3 iterations.  
To avoid these problems, it is advisable to use only the Register Tool to change the configuration while all applications being changed have been closed.

### 4.3 Pause/ Continue

If automatic registration of the certificates on the token is not desired, you can pause the Register Tool. Select Pause from the context menu of the system tray icon to temporarily stop automatic registration. Once paused, you can select Continue from the same menu.



In addition to adding the certificates to the user store, they can also be added to the machine store. Set the registry value at:

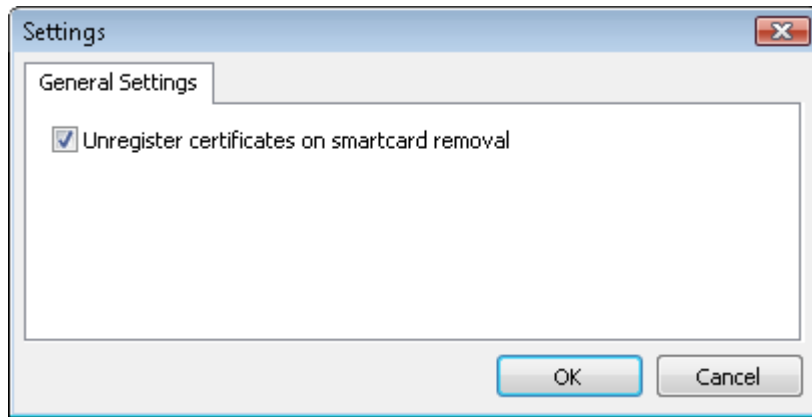
[HKEY\_LOCAL\_MACHINE\SOFTWARE\WISEkey\smart security interface]

To store certificates in the machine store, set "CSP\_RegisterMachineStore"=hex:01.

Revert the value to hex:00 to disable storing the certificates in the machine store.

### 4.4 Settings

The default functionality of the Register Tool is to register certificates automatically as soon as a token is inserted. Once the token is removed the certificates can be unregistered automatically. If this is desired, you can configure this using "Settings".



An alternate way of accessing this option is modifying the registry entry in

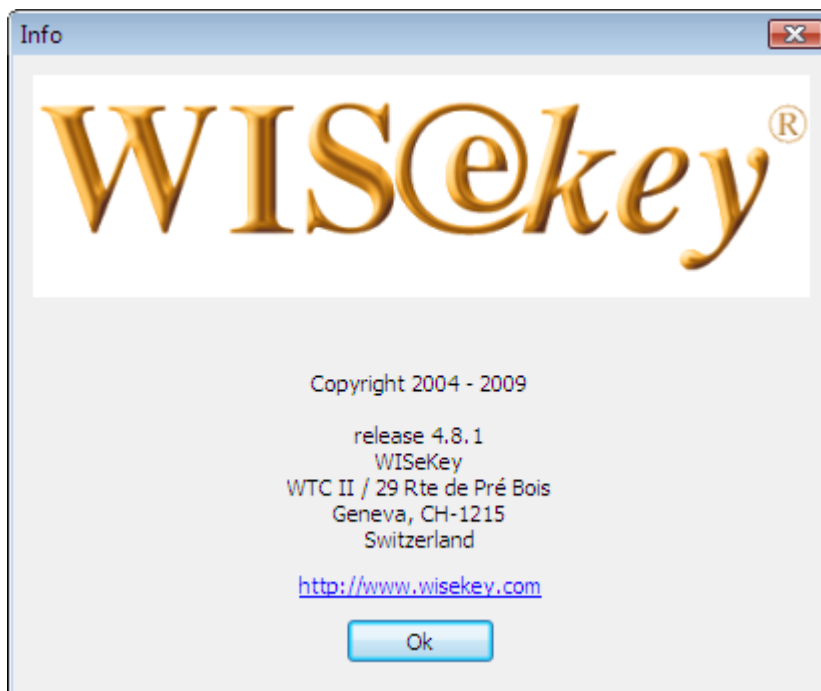
[HKEY\_LOCAL\_MACHINE\SOFTWARE\WIS@key\smart security interface]

To deactivate automatic unregistering, use: "CSP\_DeactivateUnregister"=hex:00.

To activate automatic unregistering, use: "CSP\_DeactivateUnregister"=hex:01.

## 4.5 About

For information about the version of the Register Tools and the manufacturer, select "About" in the menu of the tray icons:

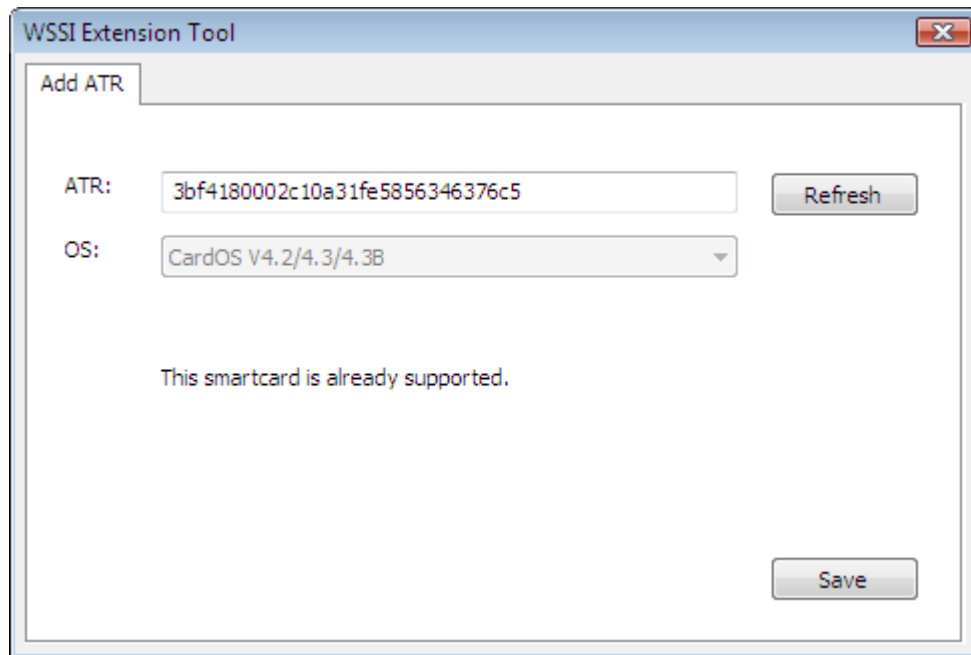


## 4.6 Exit

With "Exit" in the menu of the tray icon you can end the Register Tool.

## 5 WIS@key Extension Tool

The WIS@key Extension Tool can be used to associate smart card operating systems with new ATRs. Without a valid association, correct operation of the smart card cannot be guaranteed.



Follow these steps to make a new ATR/Card OS association:

1. Insert the smart card into the reader.  
The card ATR is displayed in the upper field.
  - a. If an OS is associated with the ATR, the OS field is locked and cannot be changed while the card is in the reader.
  - b. If no OS is associated with the ATR, select the correct OS or a close as possible match.
2. Press "Save" to store the information.

If the actual Operating System on the card is either unknown or not available, select one that matches the OS most closely, e.g. select the generic "JCOP" OS entry if the exact JCOP xx version number is not known.

**Note: On Windows Vista, you must run this tool as administrator.**

## 6 CSP of WIS@key Smart Security Interface

The Windows operating system supports cryptographic functionalities like encryption and digital signature by the so-called Crypto-API. Furthermore, CSPs (Cryptographic Service Providers) enable programs to support smart cards. During the installation of **WIS@key Smart Security Interface** the **WIS@key Smart Security Interface-CSP** (in short wkCSP) has been added.

Using wkCSP enables a number of programs and functions that come with a Windows Operating System, like Outlook Express, Internet Explorer, network login and VPN-login to use smart-cards and USB tokens. They will be explained in the following.

**NOTE:** Here, you will not find a description how to configure your Microsoft environment for the use of smart cards or USB tokens. Please consult the help files for Outlook Express and the Internet Explorer. To configure the network login and the VPN-login for smart cards please consult the documentation of the Windows 2000 Server.

### 6.1 General Proceedings

In order to use a Microsoft product in connection with the CSP for the first time on a certain computer the certificates on the token must have been registered. Please refer to chapter 4 "Register Tool" and 4.9.9 "Function "Register Certificate"" if you want to learn more about how to register your certificate.

The token must contain keys and certificates. There are several different possibilities to obtain these.

- Generating a key pair and corresponding certificate directly on the smart card with the functions of standard browsers, like Internet Explorer or Netscape. Thereby the token is accessed using the modules of **WIS@key Smart Security Interface**, i.e. correspondingly by wkCSP or wkP11. Point your browser to `http://<Servername of Enterprise-CA>/certsrv` if a certificate server is available.
- Import of existing keys and certificates on the smart card which were generated by other CAs or trust centers.

**Note:** If you request a certificate from a trust center, you might be requested to choose a security module, e.g. a token. In this case choose the corporate profile, the wkCSP or the wkP11. Furthermore, your smart card has to be inserted in the card reader, so that certificates can be written on it.

The programs must be configured to work with your keys and certificates. Some programs require root-certificates to be installed in certain directories, others require registering the certificate.

In the following chapters only the special features of the corresponding application will be explained.

### 6.2 Smart Card Login to a Windows 2000 Domain

The following is a brief outline of the steps involved in setting up smart card or USB token login.

- Setup of ADS. Please ensure the correct configuration of the DNS-Server.
- Installation of the Enterprise CA and at least the templates "Enrollment Agent", "Smartcard Logon" and "Smartcard User".
- Then an Enrollment-Agent-Certificate must be generated and registered on the computer where the smart cards should be personalized.
- After that, the smart cards for users may be issued over the Enrollment Station.

### 6.3 SSL- Authentication with Smart Card over the Internet Explorer

To use certificates stored on hardware tokens for SSL connections, the certificate must have been registered with the Windows Certificate Store. This can be done using user edition of WSSI and the Register Tool (refer to chapter 4 "Register Tool").

#### **6.4 Outlook Express with Electronic Signature and Encryption via Smart Card**

Electronic signing and encryption requires the certificates to be registered the same as for SSL connections. Once this is done, the desired certificate for signing and encryption can be chosen from "Tools → Accounts → E-Mail → Preferences → Security".

Normally, there are pull-down menus in the email windows that you may click encryption and/or signing an email in order to use the security functionalities. The verification of incoming signed emails for instance uses the red "signed" symbol in the right corner of the email window

In order for Outlook Express to automatically recognize the right key and corresponding certificate, the certificate should lie in the address book, i.e. the certificate should be imported into the "Digital IDs". Highlight the name in the address book and choose the tab "Digital IDs" over the context menu. On this tab you can import the certificate for the chosen contact.

#### **6.5 Windows VPN-Login with Smart Card**

You should generate keys and certificates with the Microsoft Enterprise-CA. Furthermore the certificate must be registered.

## 7 PKCS#11-Module of WISekey Smart Security Interface

The use of software that supports PKCS#11 is enabled by the **WISekey Smart Security Interface-PKCS#11** (abbreviated wkP11). The matter of applications and functionalities with tokens like network login, SSL, email security with Netscape and other producers are explained briefly.

**NOTE:** *There are no description on how to configure each environment to use wkP11. If your application is not covered here, please consult the corresponding documentation that comes with the application.*

**IMPORTANT:** *The PKCS#11 module is a DLL by the name "wkP11.dll" and is installed in the system directory. Usually this is C:\Windows\system32.*

**Remark:** Despite strict measures for the quality of PKCS#11 modules by the different manufacturers, WISekey gmbh can not guarantee the compatibility with each PKCS#11 Module of a third party manufacturer.

### 7.1 General Methodology

In the following some general notes are made for the employment of wkP11. General precondition is the installation of wkP11. This will be installed automatically along with **WISekey Smart Security Interface**.

There are several different ways of obtaining certificates which are described in section 8.1 "General Proceedings".

As described in chapter 4 "Register Tool" the PKCS#11 module can be installed from the WISekey user interface. Alternatively, it is possible to install the module manually with the help of the file "registerPKCS11.html" and uninstall with the help of the file "unregisterPKCS11.html". Both files are located in the installation directory of the WSSI. The installation directory defaults to

```
c:\program files\WISekey\smart security interface xx\
```

In the following chapters only the special features of the respective application will be explained.

### 7.2 Smart Card Login to a Novell eDirectory (formerly NDS)

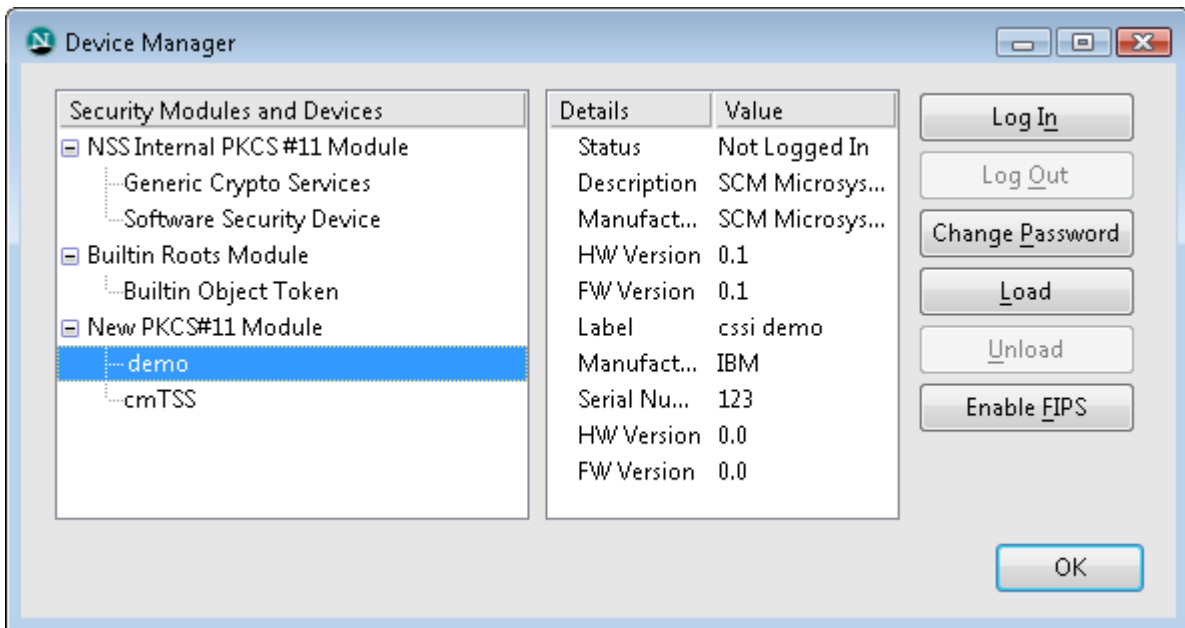
This requires a very good understanding of the administration of Novell servers. To enable smart card or USB token login to an eDirectory you explicitly need the product NMAS and the corresponding Universal Smartcard Login Method.

### 7.3 SSL- Authentication with Smart Card over Netscape

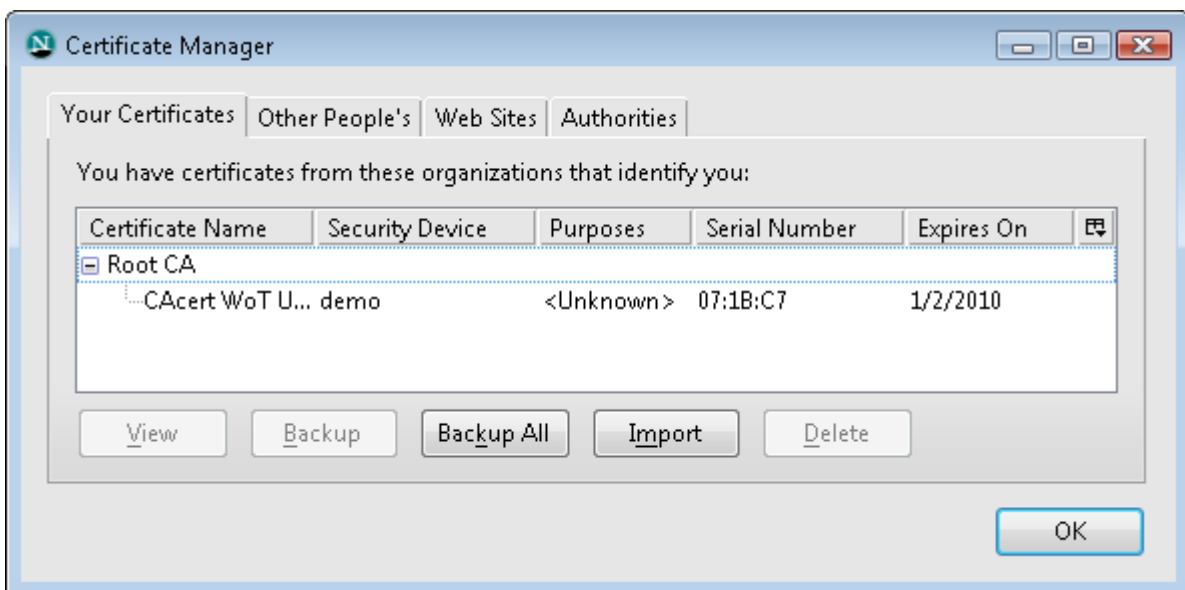
The notes for the configuration of Netscape are presented by example of version 7.

**Example:** Netscape 7.01

You can call "Manage Security Devices" in Netscape 7.01 via "Edit"→"Preferences"→"Privacy & Security"→"Certificates". From here you can load the wkP11, so that SSL and emails can be used with tokens.



Furthermore, you can call the Certificate Manager of Netscape from the same tab by clicking "Manage Certificates..."



### 7.4 Email-Security by Smart Cards and Tokens with Netscape’s Messenger

The notes for the employment of Netscape and screen shots to manage certificates and modules are available in the example of version 7 in the previous section.

Normally, there are pull-down menus in the email windows, where you can tick whether an email should be encrypted and/or signed. Functions for verification of received signed emails and decryption are available as well.

## 8 References

[PKCS#5] <http://www.rsasecurity.com/rsalabs/pkcs/index.html>

[PKCS#11] <http://www.rsasecurity.com/rsalabs/pkcs/index.html>

[MS\_CA] HOW TO: Configure a Certificate Authority to Issue Smart Card Certificates in Windows 2000: <http://support.microsoft.com/default.aspx?scid=kb;en-us;Q313274&sd=tech>

Guidelines for Enabling Smart Card Logon with Third-Party Certification Authorities:  
<http://support.microsoft.com/default.aspx?scid=kb;en-us;Q281245>

[MS\_SC] Windows 2000 Server Documentation, Smart card Administration:  
[http://www.microsoft.com/windows2000/en/server/help/default.asp?url=/windows2000/en/server/help/sag\\_SC\\_admin.htm](http://www.microsoft.com/windows2000/en/server/help/default.asp?url=/windows2000/en/server/help/sag_SC_admin.htm)

## 9 Information / Export Restrictions

WIS@key SA  
WTC II / 29 Rte de Pré Bois  
Geneva, CH-1215  
Switzerland

Manual Revision: November 3, 2009

### © Copyright WIS@key SA 2009

All rights reserved. Without the express prior written consent of WIS@key you must not distribute, edit or translate copyrighted material.

### Trade Mark

All mentioned software and hardware names are in most of the cases trade marks and are liable to legal requirements.

### Please observe!

**The product delivered to you is liable to export control. Please observe the legal requirements of specific countries. For export out of the EU an export approval is necessary.**

## Appendix A: Reference for Developers

In this appendix there are detailed specification regarding the supported functions of the PKCS#11-standard, a synopsis of particular functions and a list of objects and mechanisms. This information is useful and necessary for application developers who want to develop their own applications supporting the wkP11.

### Functions according to PKCS#11-Standard

In the following there are three lists of functions according to PKCS#11-Standard. The lists are supported, incompletely supported, and not supported functions by **WIS@key Smart Security Interface**:

Supported Functions	Incompletely Supported Functions / Deviations	Not supported functions
C_CancelFunction <sup>a</sup>	C_GetObjectSize	C_CopyObject
C_CloseAllSessions	C_GetTokenInfo	C_DecryptDigestUpdate
C_CloseSession	C_Initialize	C_DecryptVerifyUpdate
C_CreateObject	C_OpenSession	C_DeriveKey
C_Decrypt	C_SignRecover <sup>b</sup>	C_DigestEncryptUpdate
C_DecryptFinal	C_SignRecoverInit <sup>c</sup>	C_DigestKey
C_DecryptInit	C_WaitForSlotEvent	C_GetFunctionStatus
C_DecryptUpdate		C_GetOperationState
C_DestroyObject		C_SeedRandom
C_Digest		C_SetOperationState
C_DigestFinal		C_SignEncryptUpdate
C_DigestInit		
C_DigestUpdate		
C_Encrypt		
C_EncryptFinal		
C_EncryptInit		
C_EncryptUpdate		
C_Finalize		
C_FindObjects		
C_FindObjectsFinal		
C_FindObjectsInit		
C_GenerateKey		
C_GenerateKeyPair		
C_GenerateRandom		
C_GetAttributeValue		
C_GetFunctionList		
C_GetInfo		
C_GetMechanismInfo		
C_GetMechanismList		
C_GetSessionInfo		
C_GetSlotInfo		
C_GetSlotList		
C_InitPIN		
C_InitToken		
C_Login		
C_Logout		
C_SetAttributeValue		
C_SetPIN		
C_Sign		

<sup>a</sup> returns CKR\_FUNCTION\_NOT\_PARALLEL

<sup>b</sup> use C\_Sign

<sup>c</sup> use C\_SignInit

C\_SignFinal  
 C\_SignInit  
 C\_SignUpdate  
 C\_UnwrapKey  
 C\_Verify  
 C\_VerifyFinal  
 C\_VerifyInit  
 C\_VerifyRecover  
 C\_VerifyRecoverInit  
 C\_VerifyUpdate  
 C\_WrapKey

---

## Synopsis of specific functions

### C\_Finalize

Parameter: pReserved (CK\_VOID\_PTR)  
 Description: Sessions will be closed.  
 Slots will be closed.  
 Reserved Memory will be freed.  
 Deviation: pReserved will be ignored.  
 C\_Finalize will be called automatically on Finish.  
 If C\_Initialize is called n times in succession (without C\_Finalize in between), C\_Finalize will only be carried out after the n time.

### C\_GetObjectSize

Parameter: hSession CK\_SESSION\_HANDLE  
 hObject CK\_OBJECT\_HANDLE  
 pulSize CK\_ULONG\_PTR  
 Description: The size of an object will be returned  
 Deviation: The returned size is the minimum size of an object, which means it does not contain the size for extra attributes like label, or id. The size of private objects is the default value.

### C\_GetSlotList

Parameter: tokenPresent CK\_BBOOL  
 pSlotList CK\_SLOT\_ID\_PTR  
 pulCount CK\_ULONG\_PTR  
 Description: Returns a list of identified Slots.  
 It might occur that installed but not connected Slots will be in the list.  
 The number of Slots may be obtained by passing pSlotList a Null-Pointer.  
 If you want only the Slots with an inserted card set tokenPresent to true.

### C\_GetTokenInfo

Parameter: slotID CK\_SLOT\_ID  
 pInfo CK\_TOKEN\_INFO\_PTR  
 Description: Returns whether a card is inserted in a Slot. If the card is not inserted, CKR\_TOKEN\_REMOVED will be returned.  
 Special Feature: Inserting or removing a card from a Slot is an Event (see C\_WaitForSlotEvent). If C\_GetTokenInfo will be called the Event will be finished, even if the card was removed and C\_GetTokenInfo CKR\_TOKEN\_NOT\_PRESENT has been returned.

### C\_Initialize

Parameter: CinitArg CK\_VOID\_PTR\_PTR

Description: Library will be initialized.  
Slots will be created.  
Inserted cards are read.

Deviation: CinitArg is expected in the format CK\_C\_INITIALIZE\_ARGS. From these the flags are picked out, in particular CKF\_LIBRARY\_CANT\_CREATE\_OS\_THREADS which decides over Multi threading. The rest is ignored. If C\_Initialize is called several times, CKR\_CRYPTOKI\_ALREADY\_INITIALIZED is returned. The number is taken in account (see C\_Finalize).

### C\_InitToken

Parameter: slotID CK\_SLOT\_ID  
pPin CK\_UTF8CHAR\_PTR  
ulPinLen CK\_ULONG  
pLabel CK\_UTF8CHAR\_PTR

Description: Token will be initialized. SO pin is given by the parameter pPin, User pin will be reset to default 11111111. The maximal length of SO pin is 10 digits.

Special Feature: In Case Init (The token is empty): Card pin will be set to the same value as SO pin. After the token initialization, Card pin cannot be changed by PKCS#11, but SO pin can be changed using the function C\_SetPIN.  
In Case Re-Init (The token is already initialized): The given SO PIN will be verified firstly. Then the User PIN will be reset to default 11111111, and all objects on token will be deleted. The SO PIN and Card PIN are unchanged.

### C\_OpenSession

Parameter: slotID CK\_SLOT\_ID  
flags CK\_FLAGS  
pApplication CK\_VOID\_PTR  
Notify CK\_NOTIFY  
phSession CK\_SESSION\_HANDLE\_PTR

Description: Opens a new session on the Slot.

Deviation: Notify and pApplication are ignored and should be set to NULL\_PTR. Sessions can only be opened, if a card is inserted.

Special Feature: If a session is opened and then the card will be removed, all sessions on the Slot will return CKR\_DEVICE\_REMOVED. If there is an error with CKR\_DEVICE\_REMOVED, CKR\_TOKEN\_NOT\_RECOGNIZED or CKR\_TOKEN\_NOT\_PRESENT a pause is automatically produced on this Slot for all sessions.  
If a paused session is used again, this session will be reopened automatically.  
If a card is inserted into or removed from a Slot, then this is an Event (see C\_WaitForSlotEvent). If C\_OpenSession is called, the Event will be finished, even if the card has been removed and C\_OpenSession returned CKR\_TOKEN\_NOT\_PRESENT.

### C\_WaitForSlotEvent

Parameter: flags CK\_FLAGS  
pSlot CK\_SLOT\_ID\_PTR  
pReserved CK\_VOID\_PTR (= NULL\_PTR)

Description: flag = 0;  
The method waits until a Slot reports an Event. Then it returns the Slot with the Event in pSlot.

flag = CKF\_DONT\_BLOCK  
The method displays the Slot with Event in pSlot. If there is no Event, CKR\_NO\_EVENT will be returned.

Special Feature: If more Slots have an Event, they will be returned interchangeably. An Event persists until an access to the card occurs (e.g. by C\_OpenSession or C\_GetToken\_Info), even if an error will be returned to the card.

## Objects

All objects will be stored on the card (CKA\_TOKEN = true). Session- or other software-objects will not be supported.

The ID (CKA\_ID) indicates which objects belong together.

### CKO\_CERTIFICATE (CKC\_X\_509)

Certificate in X.509 format

Attribute	Value	Access
CKA_CLASS	CKO_CERTIFICATE	Read only
CKA_LABEL	<alias>	Read/write
CKA_VALUE	<certificate> X509Format (DER)	read/write
CKA_ID	<number>	read/write
CKA_CERTIFICATE_TYPE	CKC_X_509	read only
CKA_TOKEN	TRUE	read only
CKA_PRIVATE	FALSE	read only(**)
CKA_SUBJECT	<alias>	read only
CKA_ISSUER	<alias>	read only
CKA_SERIAL_NUMBER	<number>	read only
CKA_MODIFIABLE	TRUE/FALSE	read only(**)

(\*\*) Returns no error on trying to write.

### CKO\_PRIVATE\_KEY (CKK\_RSA)

Attribute	Value	Access
CKA_CLASS	CKO_PRIVATE_KEY	read only
CKA_LABEL	<alias>	read/write
CKA_ID	<number>	read/write
CKA_KEY_TYPE	CKK_RSA	read only
CKA_TOKEN	TRUE	read only
CKA_PRIVATE	TRUE	read only
CKA_SUBJECT	<alias>	read only(*)
CKA_SENSITIVE	FALSE	read only
CKA_DECRYPT	TRUE	read only(**)
CKA_SIGN	TRUE	read only(**)
CKA_SIGN_RECOVER	FALSE	read only(**)
CKA_UNWRAP	FALSE	read only(**)
CKA_MODULUS	Pkcs12 Format	read only
CKA_PUBLIC_EXPONENT	Pkcs12 Format	read only
CKA_PRIVATE_EXPONENT	Pkcs12 Format	not readable
CKA_PRIME_1	Pkcs12 Format	not readable
CKA_PRIME_2	Pkcs12 Format	not readable

CKA_EXPONENT_1	Pkcs12 Format	not readable
CKA_EXPONENT_2	Pkcs12 Format	not readable
CKA_COEFICIENT	Pkcs12 Format	not readable
CKA_MODIFIABLE	TRUE	read only(**)
CKA_LOCAL	TRUE	(**)(***)
CKA_START	<empty>	(***)
CKA_STOP	<empty>	(***)
CKA_EXTRACTABLE <sup>8</sup>	FALSE	read only(**)
CKA_NEVER_EXTRACTABLE <sup>2</sup>	TRUE	read only(**)

(\*) Can only be read if a corresponding certificate exists.

(\*\*) Returns no error on trying to write.

(\*\*\*) Is not supported.

#### CKO\_PUBLIC\_KEY (CKK\_RSA)

Attribute	Value	Access
CKA_CLASS	CKO_PUBLIC_KEY	read only
CKA_LABEL	<alias>	read/write
CKA_ID	<number>	read/write
CKA_KEY_TYPE	CKK_RSA	read only
CKA_TOKEN	TRUE	read only
CKA_PRIVATE	FALSE	read only
CKA_SUBJECT	<alias>	read only(*)
CKA_ENCRYPT	TRUE	read only(**)
CKA_VERIFY	TRUE	read only(**)
CKA_VERIFY_RECOVER	TRUE	read only(**)
CKA_WRAP	FALSE	read only(**)
CKA_MODULUS	pkcs12 Format	read only
CKA_PUBLIC_EXPONENT	pkcs12 Format	read only
CKA_MODIFIABLE	FALSE	read only(**)
CKA_LOCAL	TRUE	(**)(***)
CKA_START	<empty>	(***)
CKA_STOP	<empty>	(***)

(\*) Can only be read, if a corresponding certificate exists

(\*\*) Returns no error on trying to write

(\*\*\*) Is not supported

#### CKO\_DATA

##### General Data

Attribute	Value	Access
CKA_CLASS	CKO_DATA	read only
CKA_LABEL	<alias>	read/write
CKA_VALUE	<data>	read/write
CKA_TOKEN	TRUE	read only
CKA_PRIVATE	FALSE	read only(**)
CKA_APPLICATION	<alias>	read/write
CKA_MODIFIABLE	TRUE	read only(**)

(\*\*) Returns no error on trying to write.

## Mechanism

### Sign (RSA):

Description: Signs data

Order: C\_SignInit, C\_SignUpdate, C\_SignFinal  
or C\_SignInit, C\_Sign  
C\_Sign works as if C\_SignUpdate and then C\_SignFinal were called.  
C\_SignUpdate processes the data immediately.

Special Feature: Order C\_SignInit, C\_Sign(C\_SignUpdate, C\_SignFinal), C\_Sign  
(C\_SignUpdate, C\_SignFinal) where on the first C\_Sign (resp.  
C\_SignFinal) NULL\_PTR will be passed for the signature and only the  
length of the signature will be returned. The signature will be returned on  
the second C\_Sign (resp. C\_SignFinal). If C\_SignUpdate is called for the  
second time, the data must match with the data of the first time. A third call  
is not possible. For another signature C\_SignInit must be called first.

### Verify (RSA):

Description: Verifies a signature. VerifyRecover returns only the data (normally as a  
hash-value)

Order: C\_VerifyInit, C\_VerifyUpdate, C\_VerifyFinal  
or C\_VerifyInit, C\_Verify  
or C\_VerifyRecoverInit, C\_VerifyRecover  
C\_Verify works as if \_VerifyUpdate and then C\_VerifyFinal were called.  
C\_VerifyUpdate stores data only temporarily.  
C\_VerifyRecover returns the signed data.

Special Feature: Order C\_VerifyRecoverInit, C\_VerifyRecover, C\_VerifyRecover, where on  
the first C\_VerifyRecover a NULL\_PTR will be passed as data. It returns  
only the length of the data. The data will be returned on the second  
C\_VerifyRecover. A third call is not possible. For further verifications  
C\_VerifyRecoverInit must be called first.

### Encrypt (RSA):

Description: Encrypts data.

Order: C\_EncryptInit, C\_EncryptUpdate, C\_EncryptFinal  
or C\_EncryptInit, C\_Encrypt  
C\_Encrypt works as if C\_EncryptUpdate and then C\_EncryptFinal were  
called.

Special Feature: C\_EncryptUpdate stores the data temporarily. And you can pick up finished  
data with C\_EncryptUpdate. If you don't do this, you receive with  
C\_EncryptFinal all data at one time. The data is however only once avail-  
able!

### Decrypt (RSA):

Description: Decrypts data.

Order: C\_DecryptInit, C\_DecryptUpdate, C\_DecryptFinal  
or C\_DecryptInit, C\_Decrypt  
C\_Decrypt works as if C\_DecryptUpdate and then C\_DecryptFinal were  
called.

Special Feature: C\_DecryptUpdate stores the data temporarily. And you can pick up finished  
data with C\_DecryptUpdate. If you don't do this, you receive with  
C\_DecryptFinal all data at one time. The data is however only once avail-  
able!

**Digest (Hashfunctions SHA1, MD2, MD5):**

Description: A hash value is calculated from the data.

Order: C\_DigestInit, C\_DigestUpdate, C\_DigestFinal  
or C\_DigestInit, C\_Digest  
C\_Digest works as if C\_DigestUpdate and then C\_DigestFinal were called.  
C\_DigestUpdate processes the data immediately.

## Appendix B: Non-Standard Functions in PKCS#11 DLL

Two non-standard functions for the token initialization are added to the PKCS#11 library wkP11.

<b>CK_RV EraseProfile</b>	slotID	CK_SLOT_ID /* ID of the token's slot */
	pCardPIN	CK_BYTE_PTR /* CardPIN value */
	ulCardPINLen	CK_ULONG /* length of CardPIN value */
Description: Erase the existed profile on a token. In order to erase the profile, the Card-PIN must be verified.		

<b>CK_RV CreateProfile</b>	slotID	CK_SLOT_ID /* ID of the token's slot */
	pProfile	CK_UTF8CHAR_PTR /* profile name, null terminated */
	pSerNum	CK_BYTE_PTR /* serial number */
	ulSerNumLen	CK_ULONG /* length of serial number */
	pCardPin	CK_BYTE_PTR /* CardPIN value */
	ulCardPINLen	CK_ULONG /* length of CardPIN value */
	pSOPIN	CK_BYTE_PTR /* SO PIN value */
	ulSOPINLen,	CK_ULONG /* length of SO PIN value */
	pUserPIN	CK_BYTE_PTR /* UserPIN value */
	ulUserPINLen	CK_ULONG /* length of UserPIN value */
	pLabel	CK_UTF8CHAR_PTR /* 32-byte token label (blank padded) */
	ulUserPINRetry	CK_ULONG /* retry counter of UserPIN */
Description: Create a profile. The possible profile names are "CORPORATE", "PKCS15", "CNS" and "FINEID". Usually, the token must be empty or the old profile must be erased before the new profile is written to the token.		
Remark: Not all profiles are supported by all smartcards. CardOS V4.x supports: CORPORATE, PKCS15, CNS CardOS M4.0(a) supports: CORPORATE JavaCards support: CORPORATE, PKCS15, FINEID ACOS supports: CORPORATE.		

## Appendix C: Log Information

Logging information may serve to find and correct errors but impacts performance. In general, logging should be disabled. The logger should only be used by experienced users or when asked to. The log-file format is as follows: Each entry contains the function name, the parameter before and after the function call and the result of the function. Private information is hidden by a static string "[-----]", so only the length is readable.

### Convenience Files

To enable logging with the default settings .reg files can be found in the installation directory.

```
<program files>\WIS@key\smart security interface x.zz\
```

WSSI\_Param.reg contains logging parameters for PKCS#11 and the CSP.

### Registry Settings

Logging is controlled by registry entries stored in

```
[HKEY_LOCAL_MACHINE\SOFTWARE\WIS@key\smart security interface]
```

```
"LogFile_mode"=dword:00000001
```

Use 1 to enable logging, 0 to disable logging.

```
"PKCS11_LogFile_name"="c:\\temp\\wkP11.log"
```

```
"CSP_LogFile_name"="c:\\temp\\wkCSP.log"
```

```
"TSP_LogFile_name"="c:\\temp\\wkTSP.log"
```

```
"TCS_LogFile_name"="c:\\temp\\wkTCS.log"
```

Select a logging file and directory. Use only absolute paths names. Remember to maintain backslash '\' doubling.