TeleTrusT Deutschland e.V.
Non-profit organization for the Promotion of Trustworthiness
of Information and Communication Technology
www.teletrust.de

IOCTLs for Class 2 readers

Document Version: 1.14.2 release

Document Type: Specification

Project Id:

File Name: Z:\pr_ro\SW-

Author(s): Thomas Bründl, Michael Hartmann, Torsten Maykranz,
Markus Tak, Jochen Mades

Office / Company: KOBIL Systems GmbH, OMNIKEY AG Linz, SCM Microsystems

Abstract: This document describes which IOCTLs should be used within the PC/SC 1.0 architecture.

Last modified: 09.12.2004

Table Of Contents

1 Document Information

1.1 Owner / Master Location

Owner of this document is Thomas Bründl.

1.2 Change History

Version	*Author*	*Date*	*Description*
0.00.01	TB	06-02-2004	first proposal
0.00.02	MH	04/05/2004	KOBIL's extensions
0.00.03	MT	6.5.2004	Correction "FEATURE_MODIFY_PIN_FINISH" in section 2.3 define value for CM_IOCTL_GET_FEATURE_REQUEST
0.00.04	TB	7.5.2004	CM_IOCTL_GET FEATURE changed
0.00.05	MT	1.6.2004	Return code 64 02 in case of non-matching "new PIN" entries in Modify PIN operation
0.00.06	MT	16.8.2004	Return code "Pin_Operation_Aborted" for FEATURE_GET_KEY_PRESSED
1.00.01	TM	13.9.2004	Formal transfer into release version
1.00.02	TM	14.9.2004	cosmetics
1.00.03	MT	23.9.2004	Return codes for too short/long PIN entry cases
1.00.04	TB	08.11.2004	description of CM_IOCTL_GET_FEATURE improved
1.10.00	TB	17.11.2004	description of FEATURE_VERIFY_PIN_DIRECT and FEATURE_MODIFY_PIN_DIRECT; description of invalid parameter added
1.11.00	TM	09.12.2004	FEATURE_ABORT added
1.12.00	JM	09.12.2004	Description of FEATURE_MCT_READERDIRECT and FEATURE_MCT_UNIVERSAL
1.13.00	TB	10.12.2004	description of FEATURE_IFD_PIN_PROP added
1.14.00	TB	10.12.2004	change of feature values, 0x0a added for GET_KEY_PRESSED

1.3 Distribution & Approval History

Version	*Distributed to / approved by*	*Date distributed*	*Date approved*

1.4 General Information

1.5 Related higher level document

2 Definition of Features and IOCTLs

2.1 General Description

An application queries the PC/SC IFD handler via an specified IOCTL, which features it supports. As a response the application receives the IOCTL for this feature. This enables the different vendors/manufactures of the IFD handler to define their own IOCTLs.

The IOCTLs, the features and the corresponding parameters are defined as follows

2.2 GET_FEATURE_REQUEST

```
#define CM_IOCTL_GET_FEATURE_REQUEST          SCARD_CTL_CODE(3400)
```

2.3 Definition of Features

```
#define FEATURE_VERIFY_PIN_START        0x01  // OMNIKEY Proposal
#define FEATURE_VERIFY_PIN_FINISH       0x02  // OMNIKEY Proposal
#define FEATURE_MODIFY_PIN_START        0x03  // OMNIKEY Proposal
#define FEATURE_MODIFY_PIN_FINISH       0x04  // OMNIKEY Proposal
#define FEATURE_GET_KEY_PRESSED         0x05  // OMNIKEY Proposal
#define FEATURE_VERIFY_PIN_DIRECT       0x06  // USB CCID PIN Verify
#define FEATURE_MODIFY_PIN_DIRECT       0x07  // USB CCID PIN Modify
#define FEATURE_MCT_READERDIRECT        0x08  // KOBIL Proposal
#define FEATURE_MCT_UNIVERSAL           0x09  // KOBIL Proposal
#define FEATURE_IFD_PIN_PROP            0x0A  // Gemplus Proposal
#define FEATURE_ABORT                   0x0B  // SCM Proposal
```

Out:

If CM_IOCTL_GET_FEATURE_REQUEST is supported , the driver will return SCARD_SUCCESS and a TLV oriented structure which looks like.

Tag, Length (always 4 bytes),control code value for supported feature

The control code is returned in big endian format and must be used for the ScardControl function without and modification.

Following tags are currently defined

FEATURE_VERIFY_PIN_START	0x01
FEATURE_VERIFY_PIN_FINISH	0x02
FEATURE_MODIFY_PIN_START	0x03
FEATURE_MODIFY_PIN_FINISH	0x04
FEATURE_GET_KEY_PRESSED	0x05
FEATURE_VERIFY_PIN_DIRECT	0x06
FEATURE_MODIFY_PIN_DIRECT	0x07
FEATURE_MCT_READER_DIRECT	0x08
FEATURE_MCT_UNIVERSAL	0x09
FEATURE_IFD_PIN_PROP	0x0A
FEATURE_ABORT	0x0B

e.g. a driver which supports all feature will return

01 04 XX XX XX XX 02 04 XX XX XX 03 04 XX XX XX XX 04 04 XX XX XX XX 05 04 XX XX XX XX 06 04 XX XX XX XX 07 04 XX XX XX XX 08 04 XX XX XX XX 09 04 XX XX XX XX 0A 04 XX XX XX XX 0B 04 XX XX XX XX

The total length of the TLV structure can be retrieved from the lpBytesReturned paramter of the ScardControl function.

All IOCTLs are sent to the PC/SC driver of the reader by the ScardControl function. Before this IOCTLs can be used, a connection to the smart card is necessary. This can be achieved by the ScardConnect function.

2.3.1 Function Call

```
SCardControl(hCard, IOCTL_…,
        pCommand, CommandSize,
        Response, MaxResponseSize,
        &RetByteCount);
```

2.3.2 USB CCID Structure PIN Verify

```
#pragma pack(1)

typedef struct _USB_CCID_PIN_VERIFY_STRUCTURE
{
        UCHAR bTimeOut;                  //  timeout in seconds (00 means use CCID default)
        UCHAR bTimeOut2                 // timeout in seconds after first key stroke
        UCHAR bmFormatString;            //  formatting options (USB_CCID_PIN_FORMAT_xxx)
        UCHAR bmPINBlockString;          //  bits 7-4 bit size of PIN length in APDU, bits 3-0 PIN block size
                                         // in bytes after justification and formatting
        UCHAR bmPINLengthFormat;         //  bits 7-5 RFU, bit 4 set if system units are bytes, clear if bits,
                                         // bits 3-0 PIN length position in system units
        USHORT        wPINMaxExtraDigit;        //  XXYY, where XX is minimum PIN size in digits,
                                                // YY is maximum
        UCHAR bEntryValidationCondition;//  Conditions under which PIN entry should be
                                                // considered complete (see CCID)
        UCHAR bNumberMessage;                   //  Number of messages to display for PIN verification
        USHORT        wLangId;                  //  Language for messages
        UCHAR bMsgIndex;                        //  Message index (should be 00)
        UCHAR bTeoPrologue[3];                  //  T=1 I-block prologue field to use (fill with 00)
        UCHAR   ulDataLength                    // length of Data to be sent to the ICCUCHAR
        abData[1];                       //  Data to send to the ICC
        } USB_CCID_PIN_VERIFY_STRUCTURE, *PUSB_CCID_PIN_VERIFY_STRUCTURE;

#pragma pack()
```

2.3.3 USB CCID Structure PIN Modify

```
#pragma pack(1)

typedef struct _USB_CCID_PIN_MODIFY_STRUCTURE {
        UCHAR           bTimeOut;                  //  timeout in seconds (00 means use CCID default)
        UCHAR           bTimeOut2                  // timeout in seconds after first key stroke
        UCHAR           bmFormatString;            //  formatting options (USB_CCID_PIN_FORMAT_xxx)
        UCHAR           bmPINBlockString;          //  bits 7-4 bit size of PIN length in APDU, bits 3-0 PIN
                                                   // block size in bytes after justification and formatting
        UCHAR           bmPINLengthFormat;         //  bits 7-5 RFU, bit 4 set if system units are bytes clear if
                                                   // bits, bits 3-0 PIN length position in system units
        UCHAR           bInsertionOffsetOld;       //  Insertion position offset in bytes for the current PIN
        UCHAR           bInsertionOffsetNew;       //  Insertion position offset in bytes for the new PIN
        USHORT          wPINMaxExtraDigit;         //  XXYY, where XX is minimum PIN size in digits,
                                                   // YY is maximum
        UCHAR           bConfirmPIN;               //  Flags governing need for confirmation of new PIN
        UCHAR           bEntryValidationCondition;//  Conditions under which PIN entry should be
                                                           // considered complete
        UCHAR           bNumberMessage;            //  Number of messages to display for PIN verification
        USHORT                  wLangId;           //  Language for messages

        // Only so many bMsgIndex fields are present as are indicated by bNumberMessage

        UCHAR           bMsgIndex1;                //  Index of 1st prompting message
        UCHAR           bMsgIndex2;                //  Index of 2d prompting message
        UCHAR           bMsgIndex3;                //  Index of 3d prompting message

        // The remaining fields of this structure are at an offset which depends on how many message
        // indices appear. Hence, this structure is only an accurate mapping of the command when
        // bNumberMessage == 3...

        UCHAR           bTeoPrologue[3];           // T=1 I-block prologue field to use (fill with 00)
        UCHAR    ulDataLength                      // length of Data to be sent to the ICC          UCHAR
        abData[1];                 // Data to send to the ICC
        } USB_CCID_PIN_MODIFY_STRUCTURE , *PUSB_CCID_PIN_MODIFY_STRUCTURE ;

#pragma pack()
```

2.3.4 MCT-Structures

```
#pragma pack(1)
typdef struct _MCTUniversal
{
	UCHAR SAD;
	UCHAR DAD;
	USHORT BufferLength;
	UCHAR buffer[1];
} MCTUniversal_t, *PMCTUniversal_t;
#pragma pack()
```

2.3.5 USB CCID Structure PIN Properties

```
#pragma pack(1)

typedef struct _USB_CCID_PIN_PROPERTIES
{
  USHORT wLcdLayout;                     // display characteristics as defined in [4]
  UCHAR bEntryValidationCondition;       // bitmap as defined in [4]
  UCHAR bTimeOut2;                       // 0 = IFD does not distinguish bTimeOut from bTimeout1,
                                         // 1 =  IFD does distinguishes bTimeOut from bTimeout1,
} USB_CCID_PIN_PROPERTIES, *PUSB_CCID_PIN_PROPERTIES;

#pragma pack()
```

2.3.6 FEATURE_VERIFY_PIN_START / FEATURE_MODIFY_PIN_START

FEATURE_VERIFY_PIN_START uses the USB CCID Structure PIN Verify for the input buffer

FEATURE_MODIFY_PIN_START uses the USB CCID Structure PIN Modify for the input buffer.

The driver will return no data back to the ScardControl function.

In case of a parameter error of the passed structure the ScardControl may return ERROR_INVALID_PARAMETER (0x57).

2.3.7 FEATURE_GET_KEY_PRESSED

After the IOCTL for FEATURE_VERIFY_PIN_START or FEATURE_MODIFY_PIN_START has been sent to the driver, the IOCTL for FEATURE_GET_KEY_PRESSED can be used to determine if a key has been pressed.

The PC/SC driver will send one byte back to the ScardControl function according following table.

‘0’– ‘9’ button (valid keys)	0x2B
Cancel button	0x1B
Correction/Backspace button	0x08
Enter/Ok button	0x0d
No key since last call	0x00
Pin_Operation_Aborted	0x40 (used e.g. for timeout indication, parameter error)
all keys cleared (by backspace)	0x0a (This value can be returned optionally)

Applications can use the returned information for displaying a feedback for user (e.g. Pseudodisplay). It must use this IOCTL in a polling mode.

2.3.8 FEATURE_VERIFY_PIN_FINISH / FEATURE_MODIFY_PIN_FINISH

The IOCTL for the FEATURE_VERIFY_PIN_FINISH or FEATURE_MODIFY_PIN_FINISH must be used to retrieve the final result from the SPE operation..

The driver will return 2 bytes according to following table :

64 01	SPE operation was cancelled by the 'Cancel' button
64 00	SPE operation timed out
64 02	Modify PIN operation failed because two "new PIN" entries do not match
64 03	User entered too short or too long PIN regarding MIN/MAX PIN Length Note: as this error code is not known by CT-API implementations, it should be mapped to 64 01 on CT-API level.
6b 80	invalid parameter in passed structure
SW1 SW2	result from the card

4)64 02 occurs if the user enters two different "new PIN"'s during modify PIN operation. In that case, no change PIN command has been sent to the smart card

2.3.9 FEATURE_VERIFY_PIN_DIRECT / FEATURE_MODIFY_PIN_DIRECT

The IOCTLs for these features use the same USB CCID structures which are already described above.

The main purposes for these IOCTLs are

1) class 2 readers without for which no user feedback is required, wanted or supported f. ex. serial readers
2) class 2 readers with display which is used for SPE messages and user feedback

2.3.10 FEATURE_ABORT

This IOCTL FEATURE_ABORT can be used to cancel any of the actions initiated by the following IOCTLs:

#define FEATURE_VERIFY_PIN_START

#define FEATURE_MODIFY_PIN_START

The IOCTLs FEATURE_VERIFY_PIN_DIRECT and FEATURE_MODIFY_PIN_DIRECT will NOT require this abort IOCTL.

This IOCTL is required for devices which do not have a display, as end users may use a CANCEL button at the host application to cancel the SPE.

This IOCTL will always succeed. Also, after this ABORT IOCTL, there is no need for applications to call the IOCTL FEATURE_VERIFY_PIN_FINISH respectively FEATURE_MODIFY_PIN_FINISH, as the same 2 bytes as returned by finish can be returned by this ABORT IOCTL.

The driver will return 2 bytes according to following table :

64 80	SPE operation was aborted by the a ‘Cancel’ operation at the host system
SW1 SW2	result from the card

2.3.11 FEATURE_MCT_READERDIRECT

This IOCTL FEATURE_MCT_READERDIRECT can be used to transmit a command to the cardterminal.

The driver will return a buffer containing data (data size can be null) and two status bytes SW1 SW2 according to table 12 of [1] and chapter 4.2 of [3].

2.3.12 FEATURE_MCT_UNIVERSAL

This IOCTL FEATURE_MCT_UNIVERSAL can be used to transmit a command to the cardterminal or to any of the ICCs of the terminal.

Therefore a MCTUniversal struct is passed to the driver. The SAD and DAD fields have to be fulfilled with values concerning to table 6 of document [2]. The Buffer field contains the command APDU.

The driver will return a MCTUniversal struct with SAD and DAD fields containing values concerning to table 7 of [2].

The Buffer field will contain response data and two status bytes SW1 SW2 according to table 12 of [1] and chapter 4.2 of [3].

2.3.13 FEATURE_IFD_PIN_PROPERTIES

This feature can be used – if supported by the IFD – to retrieve the properties of the IFD regarding PIN handling :

1) The IFD returns the size of a possible display as described in [4]
2) The IFD returns which entry validation conditions are supported as descriped in [4]
3) The IFD returns if it distinguishes between bTimeOut from bTimeOut2

The input parameter for this feature is a NULL pointer.

The output parameter is a pointer to a _USB_CCID_PIN_PROPERTIES structure.

3 References

[1] International technology – Identification cards – Integrated circuit(s) cards with contacts – Part 4: Interindustry commands for interchange; International Standard ISO/IEC 7816-4:1995(E)

[2] Part 3 - Application Independent Card Terminal Application Programming Interface for ICC Applications (CT-API 1.1); Deutsche Telekom AG / PZ Telesec, SIT Fraunhofer Institut für Sichere Telekooperation, TÜV Informationstechnik GmbH, TeleTrust Deutschland e.V.; 1998

[3] Part 4 - Application Independent Card Terminal Application Programming Interface for ICC Applications (CT-API 1.1); Deutsche Telekom AG / PZ Telesec, SIT Fraunhofer Institut für Sichere Telekooperation, TÜV Informationstechnik GmbH, TeleTrust Deutschland e.V.; 1998

[4] USB Serial Bus Device Class Spec for USB Chip/Smart Card Interface Devices revision 1.00