Open Supervised Device Protocol (OSDP)
Version 2.1.5

Communication Protocol for Peripheral Devices
with Data Security Extension

September 2012

Foreword

This document, OSDPv2.1.5, is maintained by the SIA Standards Access Control and Identity Subcommittee. As with many specifications, SIA anticipates that there may be questions, interpretations, and extensions that may arise when using this specification. Please send all correspondence of this nature to osdp@siaonline.org. This address will be monitored by SIA staff and all correspondence will be forwarded to the attention of the SIA Standards Access Control and Identity Subcommittee.

Patent Information Clause

SIA draws attention to the fact that it is claimed that adherence with this document may involve the use of the following patents and the foreign counterparts:

1. United States, 6575360, Device and method for personalizing chip cards
2. United States, 7853789, Method and system for establishing a communications pipe between a personal security device and a remote computer system

SIA takes no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured SIA that they are willing to offer licenses under reasonable and non-discriminatory terms and conditions with applications throughout the world. In this respect, the statements of the holders of these patent rights are registered with SIA Information may be obtained from:

ActivIdentity Inc., 6623 Dumbarton Circle Fremont, CA 94555 United States

http://info.hidglobal.com/ai-transparent-mode-license.html

Table of Contents

Revision History

2012/09/28
- Marked "version 2.1.5"
- Replaced contradicting incidences of REPLY DELAY and REPLY TIMEOUT
- Recommend CRC method for new devices
- Marked Section 4.1.3 as Obsolete
- Expanded PD Busy Reply definition
- Added Blue color value in Section 4.10

2012/03/21
- Marked "Version 2.1.4"
- Update the secure messaging protocol to exclude cmd/reply from the encrypted portion
- Mandate either CheckSum or CRC even when the message is sent over secure channel (i.e. has a MAC)
- Clarifications on when CheckSum/CRC is invalid & security conditions are not satisfied

2012/03/05
- Marked "Version 2.1.3"
- Introduce PD busy Reply message
- Secure messaging cleanup/clarifications/diagrams
- Test vectors

2012/02/29
- marked "Version 2.1.2"
  Removed the "Smart Card Specific" commands and replaced them with alternate messages in a new appendix, **Appendix E**. Messages removed from the main body: osdp_XMIT, osdp_RMODE, osdp_SPE, osdp_SCDONE, osdp SCREP, osdp_PRES, and osdp_SPER.
  - Moved paragraph addressing cks/crc error handling from 2.7 to 3.7 and changed the recommended behavior to send osdp_NAK.
  - defined osdp_NAK error code 0x01 as bad cks/crc/mac[4]

2011/09/02
- Appendix D: modified the "Notes" section of the osdp_KEYSET command,
  - added value assignment table for SCS_xx codes
  - added a definition for "Padding", updates MAC, Wrap, and Unwrap accordingly
  - In 2.7, increased the max REPLY DELAY from 50 ms to 200 ms (missed earlier)

2011/08/11
- updated the Copyright list to include the three main contributors: Merc, HID, & Codebench
  In 2.4, increased the max REPLY DELAY from 50 ms to 200 ms

2011/06/28
- numerous updates: expanded description of the message header components: CTRL::CKSUM/CRC and MULTI; Security Block
  Expanded structures for osdp_BIOREAD, osdp_BIOMATCH, osdp_BIOREADR, and osdp_BIOMATCHR
  Revised Appendix D

2011/03/17
- minor updates to Secure PIN Entry description

2010/11/12
- added smart card commands and replies
- removed Reply Status Field
- updated Appendix C
- reformatted document (Codebench)

2009/08/14
- defined NAK codes 5 and 6 for Reply 0x41
- Added Appendix C "Preliminary" encryption extension specifications
- Added encryption support commands and replies

2009/02/07 - Added the Data Transfer command – 0x6F
2009/01/13 - Extended the usage specification of "temp text time" in Command 0x6B
2007/03/07 - Changed the name of the protocol from "Pdp-1" to OSDP. It stands for "Open Supervised Device Protocol"
2007/01/26 - Defined address = 0x7F for "broadcast" support mode, (HID/MM)
- Default communication address and baud rate assignment recommendations (HID/MM)
- Command 0x6E and Reply 54: communication address and baud rate configuration (HID/MM)
- Command 0x80 and Reply 0x90 "pass-through" messages (HID/MM)
- Updated Reply 0x53, showing key encoding guidelines

1 Introduction

This document describes the communication protocol for interfacing one or more Peripheral Devices (PD) to a Control Panel (CP). This document specifies the protocol implementation over a two-wire, multi-dropped, serial communication channel, such as RS-485. This protocol is extensible to allow transport over other media, such as TCP/IP.

Communication Settings

2.1 Physical Interface

Half-duplex RS-485 - one twisted pair, shield/signal ground

2.2 Signaling

Half duplex asynchronous serial

8 data bits

1 stop bit

no parity bits

9600, 19200 or 38400 Baud suggested

2.3 Character Encoding

The complete 8-bit character is used. All possible bit patterns may appear within a message.

2.4 Channel Access

The communication channel is used in the "interrogation/reply" mode. Only the CP can spontaneously send a message, and each message it sends is addressed to one and only one PD. The PD shall send a single reply message to each message addressed to it. If the PD has to wait on the user, other hardware, or perform computations that would take more time than the REPLY DELAY defined in section 2.7, the PD will send an osdp_BUSY reply. The actually data-laden reply must wait for the next osdp_POLL command from the CP. If the reply data requires multiple packets, the CP must osdp_POLL for each packet of the reply.

2.5 Multi-byte Data Encoding

Messages are constructed using a character stream model, meaning that all data shall be packed without any "alignment pad" characters.

Numeric data types that require more than 1 byte are stored with the least significant byte first ("little-endian" form).

2.6 Packet Size Limits

The implementation of the standard message set requires all devices to be able to accept packets up to 128 bytes long, and be able to tolerate messages addressed to other devices having a total length not exceeding 1440 bytes.

Note: this protocol's primary purpose is to support communication to simple devices on a shared (multi-dropped) channel. Large packets should be avoided. Data blocks exceeding the packet limits must be sent in multiple segments as detailed in on Page 4.

A recommendation for PDs that perform extended capabilities, such as Behavior Profiles

described in Appendix E, is that they should adjust their receive packet size capabilities to allow the transmission of its supported commands in a single packet.

2.7 Timing

The transmitting device shall guarantee a gap of a minimum of two character times before it may access the communication channel. This idle line delay is required to allow for signal converters and/or multiplexers to sense that the line has become idle.

The transmitting device shall drive the line to a marking state for a minimum of 1 character time before starting to send the first character of a message. (This can be achieved by sending a character with all bits set to '1'.) A new message may not begin any sooner than full character time following the last bit of the last message.

The transmitting device shall stop driving the line no longer than one full character time after the transmission of the stop bit of the last character of a message.

A device must begin the transmission of its reply in less than ***the defined REPLY_DELAY*** from the last character of the message requesting the reply.

The REPLY DELAY shall not exceed 20 milliseconds. The REPLY DELAY is defined as the time measured from the receipt of the checksum character of the command to the transmission of the first byte of the reply. The typical REPLY DELAY should be less than 3 milliseconds. If a device is overwhelmed it can send a BUSY message(s) (The longest REPLY DELAY occurs when local data is being processed, typically an infrequent event.)

The PD shall consider its communication "off-line" if the period between messages to which it responds exceeds 8 seconds. Both sides must reset the connection state to “off-line” and reinitiate a new connect sequence.

If the CP does not receive a reply before the REPLY_DELAY, it re-sends the last command.

2.8 Message Synchronization

The general procedure for a peripheral device (PD) to obtain message synchronization is to wait for an inter-character timeout then look for a Start-Of-Message (SOM) code. The device should then receive and store at least the header fields while computing the checksum/CRC on the rest of the message. If the checksum is good, only the PD that matches the address field processes the message. All other PDs, however, should monitor the packet by counting the remaining portion of packet to be able to anticipate the start of the next packet.

While receiving or monitoring a packet, an inter-character timeout, or a data format inconsistency shall abort the receive sequence. Once aborted, the PD should re-sync using the method described above.

The nominal value of the inter-character timeout shall be 20 milliseconds. This parameter may need to be adjusted for special channel timing considerations.

Packet Format

All messages, regardless of origin, share the same structure.

Byte	Name	Meaning	Value
0	SOM	Start of Message	0x53
1	ADDR	Physical Address of the PD	0x00 – 0x7E 0x7F = broadcast
2	LEN_LSB	Data Length Least Significant Byte	Any

Byte	Name	Meaning	Value
3	LEN_MSB	Data Length Most Significant Byte	Any
4	CTRL	Message Control Information	See List
	SEC_BLK_LEN	(optional) Length of Security Control Block	Any
	SEC_BLK_TYPE	(optional) Security Block Type	See List
	SEC_BLK_DATA	(optional) Security Block Data	Based on type
	CMND/REPLY	Command or Reply Code	See List
	DATA	(optional) Data Block	Based on CMD/REPLY
	MAC [0]	Present for secured messages	
	MAC [1]		
	MAC [2]		
	MAC [4]		
	CKSUM/CRC_LSB	Checksum, or, CRC-16 Least Significant Byte	
	CRC_MSB	(optional) CRC-16 Most Significant Byte	

3.1 SOM – Start of Message

The constant value 0x53 (93 decimal), begins each message header. This character is used for synchronization.

3.2 ADDR – Address

The 7 least significant bits of this character represent the address of the PD to which the message is directed, or the address of the PD sending the reply. A replying PD must also set the most significant bit (0x80) in the address field.

Address 0x7F is reserved as a special "BROADCAST" address that each PD will accept and respond to, just as if it matched its communication address. The reply message will use 0x7F plus the reply flag (0x7F+0x80=0xFF) in its address field. Since each PD will respond to 0x7F, the use of the broadcast address should be limited to controlled (single PD) configurations.

3.3 LEN – Length

The value of the two-character length field is the total number of characters contained in the message, including the SOM through the CKSUM or CRC characters.

3.4 CTRL - Control

BIT	MASK	NAME	Meaning
0-1	0x03	SQN	The sequence number of the message is used for message delivery confirmation and for error recovery.
2	0x04	CKSUM/ CRC	Set – 16-bit CRC is contained in the last 2 bytes of the message Clear – 8-bit CHECKSUM is contained in the last byte of the message
3	0x08	SCB	Set – Security Control Block is present in the message

BIT	MASK	NAME	Meaning
			Clear – No Security Control block in the message
4-6	0x70		Deprecated (formerly Reply Status Field)
7	0x80	MULTI	Set – more packets to come for this message Clear – this is the last/only packet of this message

SQN Values

The sequence number is incremented by the CP from one command to the next, skipping zero: 0->1->2->3->1->... Non-zero sequence numbers support error recovery: the Control Panel (CP) acknowledges the last reply by sending the next command with the incremented sequence number, or it repeats the command without changing the sequence number to request the repeat of the last reply. This method allows the receiver to properly handle the command: process the command if it did not receive it correctly last time (error occurred on the command), or to simply repeat the reply it already made without executing the command again (error occurred in the reply).

SQN zero should be used only for communication startup, at boot time or after a communications loss. Zero forces the PD to discard its last reply and to accept and process the current command.

CKSUM/CRC

This setting defines the message check character(s) method used to provide error detection.

The CKSUM value is the 8 least significant bits of the 2's complement value of the sum of all previous characters of the message. This mode is supported in order to allow for devices with limited resources, but new devices should use the CRC method.

The CRC calculation is applied to all previous characters of the message. Refer to Appendix C for the definition and sample code for the CRC algorithm.

Note, that during an established Secure Channel Session, the SEC_BLK_TYPE values in the Security Block will override the CKSUM/CRC bit field and the check characters are derived from a Message Authentication Code computed for the message. Refer to Appendix D for further details.

MULTI - Multi-Part Messages

For single packet messages, this bit is clear. If the size of the data payload plus the header exceeds the packet size, the sender must split the data into multiple packets and set this bit on all but the last packet. Only the first part of a Multi-Part message contains the original message code. Subsequent segments are sent with osdp_CONT as the message code.

These fragmented message packets take priority over any other data so, for instance, if the PD is sending a fragmented message when an input or badge read comes in, the PD must finish sending the fragmented message in response to osdp_POLLs from the CP. It cannot interleave the input report with the fragmented message packets.

Likewise, the CP cannot send commands to any other PD, nor can it send any command other than osdp_CONT to the PD sending the fragmented message.

3.5 Security Block

The Security Block (SB) is optional. Its presence is indicated by setting the CTRL::SBC flag. The purpose of the SB is to facilitate the implementation of data security within the OSDP framework. By itself, the SB does not define or specify the nature of the security methods used. Rather, the SB is available to support the use of various security methods as OSDP device capabilities and client

security requirements change.

See Appendix D for further details.

The Security Block (SB):

Byte	Name	Meaning	Value
0	SEC_BLK_LEN	Length of the Security Control Block	Any
1	SEC_BLK_TYPE	Security Block Type	0x01 - OSDP-sc
2 - n	SEC_BLK_DATA	Variable Length Data (optional)	Any

SEC_BLOCK_LEN

This field is set to the total byte count of the SB, including itself.

SEC_BLK_TYPE

This field defines the manner in which the security block applies to the rest of message (the optional sec_blk_data[] array , the command/reply, the optional data[] array, and the message check characters).

SEC_BLK_DATA

This section is an array whose size is (sec_blk_length-2). The data content is separately defined for each SEC_BLK_TYPE.

A PD that receives an SB, but does not support the processing of the SB should return an osdp_NAK response, error_code set to 0x05.

A PD whose settings require an encrypted connection, and receives a command without the appropriate data security extension must return an osdp_NAK response, error_code set to 0x06.

3.6 CMND/REPLY - Command/Reply Code

Commands and replies are the actual payload of the communication packets. A packet originated by the CP is called a command, and a packet returned by the PD is called a reply. The purpose and meaning of each message packet is defined by its command or reply code. The actual codes and associated data blocks (if any) are specified in detail in the following sections.

See Appendix A for a quick reference table with the numeric command values.

3.7 CHKSUM/CRC16 - Message Check Codes

OSDP supports three different forms of error detection as discussed in the CKSUM/CRC paragraph under CTRL.

If the PD does not support the indicated checksum/CRC method, or if it gets a checksum/CRC error, then PD shall send a osdp_NAK response, error_code set to 0x06.

Commands

The following commands can be sent from the CP to the PD. The value listed in the table below goes in the message CMND field. Values 0x40 through 0x8F are reserved for core commands. Values outside this range can be used for application specific and/or proprietary implementations.

4.1 Poll (osdp_POLL)

This command serves as a general inquiry. The PD may return any reply that is marked as a possible "poll response". Normally, the PD will return any unreported input data or status change information as a poll response.

Command structure: None

Reply: Any of the Reply messages marked "poll response".

4.2 ID Report Request (osdp_ID)

This command requests the return of the PD ID Report. The id request code parameter may request the extended form of the PD ID block.

Command structure: 1-byte id request code

Request Code	Meaning
0x00	Send Standard PD ID Block

Reply: osdp_PDID - ID Report

4.3 Peripheral Device Capabilities Request (osdp_CAP)

This command requests the PD to return a list of its functional capabilities, such as the type and number of input points, outputs points, reader ports, etc.

Command structure: 1-byte request code

Request Code	Meaning
0x00	Send Standard Reply

Reply: osdp_PDCAP - Device Capabilities Report

4.4 Diagnostic Function Request (osdp_DIAG)

This command controls diagnostic functions and status reports.f

Command structure: 1-byte request code

Request Code	Meaning
0x00	Default

Reply: (to be defined)

Note: Currently, this command is a placeholder.

4.5 Local Status Report Request (osdp_LSTAT)

Instructs the PD to reply with a local status report.

Command Structure: None

Reply: osdp_LSTATR - Local Status Reply

4.6 Input Status Report Request (osdp_ISTAT)

Instructs the PD to reply with an input status report.

Command Structure: None

Reply: osdp_ISTATR - Input Status Reply

4.7 Output Status Report Request (osdp_OSTAT)

Instructs the PD to reply with an output status report.

Command Structure: None

Reply: osdp_OSTATR Output Status Reply

4.8 Reader Status Report Request (osdp_RSTAT)

Instructs the PD to reply with a reader status report.

Command Structure: None

Reply: osdp_RSTATR - Reader Status Reply

4.9 Output Control Command (osdp_OUT)

The Output Control command can alter the permanent state of the output, or it can request a timed pulse output.

Command structure: 4-byte element, repeated 1 or more times

Byte	Name	Meaning	Value
0	Output Number	0 == K1, 1 == K2, etc	any
1	Control Code	Requested Output State	See below
2	Timer LSB	least significant byte, in units of 100 ms	any
3	Timer MSB	most significant byte, in units of 100 ms	any

Control Code	Meaning
0x00	NOP – do not alter this output
0x01	set the permanent state to OFF, abort timed operation (if any)
0x02	set the permanent state to ON, abort timed operation (if any)
0x03	set the permanent state to OFF, allow timed operation to complete
0x04	set the permanent state to ON, allow timed operation to complete
0x05	set the temporary state to ON, resume perm state on timeout
0x06	set the temporary state to OFF, resume permanent state on timeout

Timer values:

The timer value is specified in units of 100 milliseconds. The 16-bit value provided supports a maximum pulse time of 6,553.5 seconds, which is 1 hour, 49 minutes, and 13.5 seconds.

The PD may respond with a reply 0x4A to indicate that output(s) have changed state, or at the PD's option, it can return reply 0x40 (osdp_ACK), then send the output change report (0x4A) later.

The Output Control Command message packet may contain multiple 4-byte records. Use the total message length to determine the number of records present.

Reply: osdp_ACK, osdp_NAK, or osdp_OSTATR

4.10 Reader LED Control Command (osdp_LED)

The Reader LED Control command controls the operation of the LEDs associated with a reader. (This command supports the model where multiple LEDs may be associated with a reader.) Color and flash parameters may be specified.

Once the temporary command's timer expires the LED will revert to the last permanent state set. A timer value of zero specifies zero duration.

Command structure: 14-byte element, repeated 1 or more times

Byte	Name	Meaning	Value
0	Reader Number	0 == K1, 1 == K2, etc	Any
1	LED Number	0 == first LED	Any
Temporary Settings			
2	Control Code	The mode to enter temporarily	See Below
3	ON time	The ON duration of the flash, in units of 100 ms	Any
4	OFF time	The OFF duration of the flash, in units of 100 ms	Any
5	ON color	The color to set during the ON time	See Below
6	OFF color	The color to set during the OFF time	See Below
7	Timer LSB	least significant byte, in units of 100 ms	Any
8	Timer MSB	most significant byte, in units of 100 ms	Any
Permanent Settings			
9	Control Code	The mode to return to after the timer expires	See Below
10	ON time	The ON duration of the flash, in units of 100 ms	Any
11	OFF time	The OFF duration of the flash, in units of 100 ms	Any
12	ON color	The color to set during the ON time	See Below
13	OFF color	The color to set during the OFF time	See Below

Temporary Control Code	Meaning
0x00	NOP - do not alter this LED's temporary settings
0x01	Cancel any temporary operation and display this LED's permanent state immediately
0x02	Set the temporary state as given and start the countdown timer immediately.

Permanent Control Code	Meaning
0x00	NOP - do not alter this LED's permanent settings
0x01	Set the permanent state as given.

Color Value	Meaning
0	Black (off/unlit)
1	Red
2	Green
3	Amber
4	Blue

Notes:

The LED will flash, alternating between the color specified for ON and color specified for OFF at the rate specified by the corresponding timers. Setting both color codes to the same value will produce a steady (non-flashing) output.

The 16-bit timer applies to the temporary LED commands only.

Examples:

To cause reader-0's first LED to flash red for 3 seconds, then resume its current display mode:

0,0, 2,1,2,1,0,30,0, 0,0,0,0,0

To set the reader's second LED to display a steady green output

0,1, 1,0,0,0,0,0,0, 1,1,1,2,2

The LED Control Command message packet may contain multiple 14-byte records. Use the total message length to determine the number of records present.

Reply: osdp_ACK, osdp_NAK

4.11 Reader Buzzer Control Command (osdp_BUZ)

This command defines commands to a single, monotone audible annunciator (beeper or buzzer) that may be associated with a reader.

Command structure: 5-byte element

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	Any
1	Tone Code	Requested Tone State	See Below
2	On Time	The ON duration of the sound, in units of 100 ms	Any
3	OFF Time	The OFF duration of the sound, in units of 100 ms	Any
4	Count	The number of times to repeat the ON/OFF cycle	0 means tone continues until another tone command is received

Notes:

The Buzzer will sound, alternating between the ON and the OFF states specified by the corresponding timers. Set the OFF time to zero to produce a steady tone.

4.12 Reader Text Output Command (osdp_TEXT)

This command defines a string to be shown on a simple character oriented text display organized in a

row and column format.

Text will be written at the given starting position. If necessary, and if allowed by the command, the text may wrap to the next line.

Temporary text, if implemented, overwrites a text field for a specified time period after which the permanent text field is restored.

The "temp text time" field indicates the duration of the temp display in seconds. This field has a different meaning when used with a permanent text command. In that case, if the temp text time is zero then any active temp text shall be allowed to complete. A non-zero temp text time means that any active temp text shall be aborted and the permanent text shall be displayed immediately.

At a minimum, the PD must implement the printable ASCII character set, ranging from 0x20 to 0x7E.

Command Structure: variable-length data with a fixed 5-byte header

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	Any
1	Text Command	How to treat the text	See Below
2	Temp Text Time	The duration to display temporary text, in seconds	See Below
3	Row	The row where the first character will be displayed	1 is the top row
4	Column	The column where the first character will be displayed	1 is the left-most column
5	Text Length	Number of characters in the string	Any
6 - N	String	The string to display	Valid ASCII characters

Text Command	Meaning
0x01	permanent text, no wrap
0x02	permanent text, with wrap
0x03	temp text, no wrap
0x04	temp text, with wrap

4.13 Time and Date Command (osdp_TDSET) -- OBSOLETE

This command contains the time and date in local time.

Command structure: 7 bytes

Byte	Name	Meaning	Value
0	Year LSB		
1	Year MSB	16-bit encoding of the current year	any
2	Month	The current month	1 - 12
3	Day of Month	The current day of the month	1 - 31
4	Hour	The number of hours since midnight	0 - 24
5	Minute	The minutes past the current hour	0 - 59

Byte	Name	Meaning	Value
6	Second	The seconds past the current minute	0 - 59

Notes:

The time set command should be sent every time a new connection is established. This command should be sent as often as necessary based on the time keeping capabilities of the PD.

Reply: osdp_ACK, osdp_NAK

4.14 Communication Configuration Command (osdp_COMSET)

This command sets the PD's communication parameters. The settings will take effect AFTER the PD has completed its response to this command.

Command structure: 5 bytes

Byte	Name	Meaning	Value
0	Address	Unit ID to which this PD will respond after the change takes effect	0x00 - 0x7E
1	Baud Rate LSB	4-byte baud rate value	Any
2	Baud Rate		Any
3	Baud Rate		Any
4	Baud Rate MSB		Any

Note:

The baud rate is expressed as a 32-bit integer holding the actual value of the baud rate to use, such as 9600, 38400, etc.

If the PD is unable to comply, it will return the values that it will use after the completion of this reply.

Reply: osdp_COM – PD Communication Configuration Report

4.15 Data Transfer Command (osdp_DATA)

This command supports the transfer of data sets indexed from a reference point.

Command structure: 7-byte fixed header followed by up to 255 bytes of data

Byte	Name	Meaning	Value
0	Data Block Type	How to interpret the give data	0 – start 1 - data set indexed from address zero 9 - end of transfer
1	Block Length	Number of data bytes	Any
2	Load Address LSB	5-byte address value	Any
3	Load Address LSB	5-byte address value	
4	Load Address LSB		Any
5	Load Address LSB		Any

Byte	Name	Meaning	Value
6	Load Address MSB		Any
7 - N	Data Bytes		Any

Reply: osdp_ACK, osdp_NAK

Note:

The data block type may be extended to suit particular data transfer requirements.

*4.16 Set Automatic Reader Prompt Strings (osdp_PROMPT) *DRAFT**

This command allows the host to set the prompt strings the reader will display to prompt the user during various operations.

Command Structure: an 8-byte header followed by a variable-length string

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	any
1	Dictionary Index	Index of the dictionary to be modified	See Below
1	String Index	Index of the string to be set	See Below
2 - 6	Locale	Locale string	See Below
7	Length	Length of the following string	any
8 - N	Data	Message string in UTF-8	any

The reader has a separate dictionary of strings for different user interactions. More will be defined as needed. The String Index in the dictionary defines its meaning. The tables below show the Dictionary Index and String Index for currently defined dictionaries.

Secure Pin Entry Dictionary (0x01)	
String Index	**Default Value**
0x01	Enter PIN
0x02	Invalid PIN
0x03	PIN Accepted
0x04	Invalid PIN, Card Locked

Fingerprint Scan Dictionary (0x02)	
String Index	**Default Value**
0x01	Present Right Thumb
0x02	Present Right Index Finger
0x03	Present Right Middle Finger
0x04	Present Right Ring Finger

0x05	Present Right Little Finger
0x06	Present Left Thumb
0x07	Present Left Index Finger
0x08	Present Left Middle Finger
0x09	Present Left Ring Finger
0x0A	Present Left Little Finger

The Locale field is a string up to 5 characters long. Pad shorter strings with zeros on the end. The Local consists of an ISO 639 2 character language code followed by an optional underscore and 2 character ISO 3166 2 character territory code. For example:

en – English
en_US – English in the United States
fr – French
fr_BE – French in Belgium

Reply: osdp_ACK, osdp_NAK

4.17 Scan and Send Biometric Template (osdp_BIOREAD)

This command instructs the reader to perform a fingerprint scan and return the scan data to the CP as a poll response in osdp_BIOREADR. The type, format and quality of the scan are specified in the command structure. The reader should restore the display to its previous state when finished processing the user input.

Command Structure: 4 byte command structure

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	Any
1	BIO_TYPE	Type/body part to scan	See list below
2	BIO_FORMAT	Format of data to be returned	See list below
3	BIO_QUALITY	Quality setting for the original scan	Normalized to 0x00 - 0xFF

Biometric Types	
Value	**Meaning**
0x00	Not specified – default -
0x01	Right Thumb Print
0x02	Right Index Finger Print
0x03	Right Middle Finger Print
0x04	Right Ring Finger Print
0x05	Right Little Finger Print
0x06	Left Thumb Print
0x07	Left Index Finger Print
0x08	Left Middle Finger Print

Biometric Types	
Value	**Meaning**
0x09	Left Ring Finger Print
0x0A	Left Little Finger Print
0x0B	Right Iris Scan
0x0C	Right Retina Scan
0x0D	Left Iris Scan
0x0E	Left Retina Scan
0x0F	Full Face image
0x10	Right Hand Geometry
0x11	Left Hand Geometry

Fingerprint Formats (types 0x01 - 0x0A)	
Value	**Meaning**
0x00	Not specified – use default method to scan, then report format used
0x01	Send raw fingerprint data as a PGM 48
0x02	ANSI/INCITS 378 Fingerprint template 48

Other formats will be added as they are standardized for various biometric types.

Reply: osdp_ACK, osdp_NAK

4.18 Scan and Match Biometric Template (osdp_BIOMATCH)

If the reader supports biometric template matching, this command should be used instead of BIOREAD to improve performance. This packet instructs the reader to perform a biometric scan and to match it against the template provided in the data section of this packet and return the results to the CP as a poll response in osdp_BIOMATCHR. The reader should restore the display to its previous state when done processing the user input.

See the types and formats in Section 13.

Command Structure: 6-byte header followed by a variable length template.

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	Any
1	BIO_TYPE	Type/body part to scan	See BIOREAD
2	BIO_FORMAT	Format of attached template	See BIOREAD
3	BIO_QUALITY	Quality: threshold required for accepting the bio match	Normalized to 0x00 - 0xFF
4	BIO_LENGHT_LS	Template length, least significant byte	Any
5	BIO_LENGTH_MS	Template length, most significant byte	Any
6 - n	BIO_DATA	Array of template data of BIO_LENGHT	Any

Reply: osdp_ACK, osdp_NAK

4.19 Continue Multi-Part Message (osdp_CONT)

Instructs the PD to reply with the next packet of a multi-part message.

Command Structure: None

Reply: Any

4.20 Manufacturer Specific Command (osdp_MFG)

This command is intended to allow manufacturer specific commands to be embedded within this protocol. The data content of this command is not defined in this document beyond the following formatting guidelines:

Byte	Name	Meaning	Value
0	Vendor Code 1'st	IEEE assigned OUI, "first octet"	any
1	Vendor Code 2nd	IEEE assigned OUI, "second octet"	any
2	Vendor Code 3'rd	IEEE assigned OUI, "third octet"	any
3 - N	Data	Vendor Defined	any

Reply: osdp_ACK, osdp_NAK, osdp_MFRG – Manufacturer Specific Reply

Note:

The PD may use the Vendor Code and other content dependent values to confirm that the command contains the expected structure.

Replies

The PD must begin sending a reply less than 50 ms after it receives the last character of a valid command. If it cannot, it should send an osdp_ACK, and the actual data requested will be sent to the CP as a POLL reply when it becomes available.

If the command has a bad checksum/CRC, requests an unsupported checksum/CRC method or exceeds the PDs input buffer size, the PD should not respond. In those cases the PD cannot be sure the address field is correct, so this prevents wrong or multiple PDs from responding.

For any other detected failure, the PD should send a osdp_NAK message.

If the CP receives a packet with an invalid checksum/CRC or detects any other error, it must re-transmit the command using the same SQN as the original request.

See 2222 for a quick reference table with the numeric reply values.

5.1 General Acknowledge, Nothing to Report (osdp_ACK)

There is no reply structure associated with this reply. Sent in response to all valid commands that do not require a specific response. Also sent if the command will take more than 50ms to process, for example when user input is required.

5.2 Negative Acknowledge – SIO Comm Handler Error Response (osdp_NAK)

Reply Structure: 1-byte error code

Error Code Value	Meaning

Error Code Value	Meaning
0x01	Message check character(s) error (bad cks/crc/mac[4])
0x02	Command length error
0x03	Unknown Command Code – Command not implemented by PD
0x04	Unexpected sequence number detected in the header
0x05	This PD does not support the security block that was received
0x06	Encrypted communication is required to process this command
0x07	BIO_TYPE not supported
0x08	BIO_FORMAT not supported

5.3 Device Identification Report (osdp_PDID)

Sent in response to an osdp_ID command

Reply Structure: 12-byte fixed record

Byte	Name	Meaning	Value
0	Vendor Code 1st	IEEE assigned OUI, "first octet"	any
1	Vendor Code 2nd	IEEE assigned OUI, "second octet"	any
2	Vendor Code 3rd	IEEE assigned OUI, "third octet"	any
3	Model Number	Manufacturer's model number	any
4	Version	Manufacturer's version of this product	any
5	Serial Number LSB	4-byte serial number	any
6	Serial Number		any
7	Serial Number		any
8	Serial Number MSB		any
9	Firmware Major	Firmware revision code, major	any
10	Firmware Minor	Firmware revision code, minor	any
11	Firmware Build	Firmware revision code, build	any

Notes:

The Vendor Code is a 24-bit identifier of the manufacturer. It is recommended that each manufacturer use its IEE assigned Organizationally Unique Identifier, the same 24 bits it uses to form the MAC addresses of its ethernet-based products.

The Model Number and Version fields are assigned by and managed by the Vendor. These fields have no direct operational purpose.

The 32-bit Serial Number field is assigned and managed by the Vendor. This field has no direct operational purpose.

The Firmware Revision fields are assigned and managed by the Vendor. These fields have no direct operational purpose.

5.4 Device Capabilities Report (osdp_PDCAP)

Sent in response to an osdp_CAP command

Reply Structure: 3-byte element, repeated one or more times

Byte	Name	Meaning	Value
0	Function Code	Function/feature code	See Below
1	compliance	Level of compliance with above function	
2	Number of	Number of objects of this type	

Notes regarding field usage:

The Device Capabilities report message may contain multiple records of this form (3 bytes per record). Use the total message length to determine the number of records present.

The records may be in any order. If a function code is omitted from the list, The CP may assume that the PD has no support for that function code.

The "Function Code" directly corresponds to a defined operational capability.

The "Compliance" field indicates the extent the PD supports the Function Code. If applicable, the "Number of" field indicates that number of objects of this type that are available.

A list of Function Codes and their definition, and the corresponding compliance levels is incorporated as Appendix A of this Document.

5.5 Local Status Report (osdp_LSTATR)

Sent in response to an osdp_LSTAT command or as a "poll response"

The local status report applies to conditions directly monitored by the PD. Tamper status is detected by the PD by monitoring the enclosure tamper mechanism. Power monitor status can be derived from the status of the power supply. Normally this reply is sent in response to an osdp_POLL command if either status has changed since the last POLL.

Reply Structure: 2 status bytes

Byte	Name	Meaning	Value
0	Tamper Status	Status of tamper circuit	0x00 – normal 0x01 - tamper
1	Power Status	Status of power	0x00 – normal 0x01 – power failure

5.6 Input Status Report (osdp_ISTATR)

Sent in response to an osdp_ISTAT command or as a "poll response"

Normally, this reply is sent in response to an osdp_POLL command if the status of any of the inputs has changed since the last report. The array size is defined by the total message length.

Reply Structure: 1 status byte for each input

Status Byte Value	Meaning
0x00	Inactive

0x01	Active

5.7 Output Status Report (osdp_OSTATR)

Sent in response to an osdp_OSTAT command , an osdp_OUT command or as a "poll response"

Normally, this response is sent as a reply to an osdp_OUT command to indicate that the output(s) have changed state.

This reply can also be sent in response to an osdp_POLL if the status of any of the outputs has changed since the last report. The array size is defined by the packet length.

Reply Structure: 1 status byte for each output

Status Byte Value	Meaning
0x00	Inactive
0x01	Active

5.8 Reader Tamper Status Report (osdp_RSTATR)

Sent in response to an osdp_RSTAT command or as a "poll response"

Normally, this reply is sent in response to an osdp_POLL if the status of any of the readers has changed since the last report. The array size is defined by the total message length.

The reader tamper is applicable only in cases where an external reader is attached to the PD, and the PD is able to monitor the status of the attached reader. (Certain readers can send periodic status messages.)

Reply Structure: 1 status byte for each reader

Status Byte Value	Meaning
0x00	Normal
0x01	Not Connected
0x02	Tamper

5.9 Card Data Report, Raw Bit Array (osdp_RAW)

Sent as a "poll response"

This reply is sent in response to an osdp_POLL command after a card was read but the raw data was not decoded into a character array.

reply structure: 4-byte header, variable-length data

Byte	Name	Meaning	Value
0	Reader Number	0=="Reader 1", 1=="Reader 2"	any

Byte	Name	Meaning	Value
1	Format Code	Format of included data	0 = not specified, raw bit array 1 = P/data/P (wiegand) 2 = 5-bit/char mag 3 = 200-bit FASC-N, 32-bit HMAC, 64-bit date
2	Bit Count LSB	2-byte size (in bits) of the data at the end of the record	any
3	Bit Count MSB		any
4 - N	8 data bits	8 bits of card data MSB to LSB	any

5.10 Card Data Report, Character Array (osdp_FMT)

Sent as a "poll response"

This reply is sent in response to an osdp_POLL when decoded and formatted card data is available.

Reply Structure:3-byte header, variable-length data

Byte	Name	Meaning	Value
0	Reader Number	0=="Reader 1", 1=="Reader 2"	any
1	Read Direction	Direction of data in array	0 = forward read 1 = reverse read
2	Digit Count	Number of digits, including START, END, CKSUM	any
3 - N	data	Card data represented as a string	

5.11 Keypad Data Report (osdp_KEYPAD)

Sent as a "poll response"

This reply is sent in response to an osdp_POLL if there is any data in the keypad buffer.

Reply Structure: 2-byte header, variable-length data

Byte	Name	Meaning	Value
0	Reader Number	0=="Reader 1", 1=="Reader 2"	any
1	Digit Count	Number of keypad digits to follow	any
2 - N	data	Digits from the keypad buffer in the order in which they were entered.	See Below

The key encoding uses the following data representation:

Digits 0 through 9 are reported as ASCII characters 0x30 through 0x39
The clear/delete/'*' key is reported as ASCII DELETE, 0x7F
The enter/'#' key is reported as ASCII return, 0xD
Special/function keys are reported as upper case ASCII:
A or F1 == 0x41, B or F2 == 0x42, C or F3 == 0x43, D or F4 == 0x43,
F1 & F2 == 0x44, F2 & F3 == 0x45, F3 & F4 == 0x46, F1 & F4 == 0x47

When the reader is in Transparent mode and processing an osdp_SPE message, no key values should be added to the keypad buffer.

5.12 Communication Configuration Report (osdp_COM)

Sent in response to an osdp_COMSET command.

This reply returns the communication parameters the PD will use after sending this reply.

Reply Structure: 5-byte record

Byte	Name	Meaning	Value
0	Address	Unit ID for this PD to respond to	0x00 - 0x7E
1	Baud Rate LSB	4-byte baud rate value	Any
2	Baud Rate		Any
3	Baud Rate		Any
4	Baud Rate MSB		Any

5.13 Scan and Send Biometric data (osdp_BIOREADR)

Sent as a "poll response"

The DATA section contains the scanned result in the requested format.

Reply Structure: as defined below. Note that due to the template length, this reply may need to break up the data into multiple packets. (See the “Multi-Part Messages” paragraph.)

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	any
1	STATUS	Results of requested command	See List
2	BIO_TYPE	Bio template encoding type	See List
3	BIO_QUALITY	Scan Quality (0 = worst, 0xFF = best)	0x00 - 0xFF
4	BIO_LENGTH_LS	Template length, least significant byte	Any
5	BIO_LNGTH_MS	Template length, most significant byte	Any
6 - n	BIO_TEMPLATE	Scan image or template	Any

5.14 Scan and Match Biometric Template (osdp_BIOMATCHR)

Sent as a "poll response"

Return the appropriate CODE and 1 byte of data indicating if the scanned body part matched the biometric template sent from the host.

Reply Structure: 1-byte Reader Number, 1-byte status code followed a 1-byte result code

Byte	Name	Meaning	Value
0	Reader Number	0 == Reader-00	any
1	STATUS	Results of requested command	See List

Byte	Name	Meaning	Value
2	SCORE	Result of the biometric match	0x00 - 0xFF 0x00 – No Match 0xFF – Best Match

5.15 Manufacturer Specific Reply (osdp_MFGREP)

Sent in response to an osdp_MFGR command or as a "poll response"

This reply is intended to allow manufacturer specific messages to be embedded within this protocol. The data content of this reply is not defined in this document beyond the following:

Byte	Name	Meaning	Value
0	Vendor Code 1st	IEEE assigned OUI, "first octet"	any
1	Vendor Code 2nd	IEEE assigned OUI, "second octet"	any
2	Vendor Code 3rd	IEEE assigned OUI, "third octet"	any
3 - N	Data	Vendor Defined	any

5.16 PD Busy Reply (osdp_BUSY)

Sent in response to an osdp command if the PD is busy processing the previous command. This reply will use either checksum or CRC for message integrity even if the secure channel has been established and commands are exchanged using secure messaging. In other words, the busy reply is sent outside the secure channel and should not influence the secured messages that are sent before or after this reply.

The osdp_ACK is the appropriate response if the data requested by the command is not immediately available but will be returned in response to a subsequent osdp_POLL. Otherwise (meaning that a specific non-ACK response is required and the data is not available in time to meet the REPLY_TIMEOUT), the PD responds with osdp_BUSY until it is able to return the requested data. In this case, the CP shall continue to repeat the command in its original form until the PD returns something other than osdp_BUSY.

The sequence number of this ***reply*** will always be set to 0.

APPENDIX A - Command and Reply Code Numbers

Commands

Name	Value	Meaning	Data
osdp_POLL	0x60	Poll	none
osdp_ID	0x61	ID Report Request	Id type
osdp_CAP	0x62	PD Capabilities Request	Reply type
osdp_DIAG	0x63	Diagnostic Function Command	Request code
osdp_LSTAT	0x64	Local Status Report Request	None
osdp_ISTAT	0x65	Input Status Report Request	None
osdp_OSTAT	0x66	Output Status Report Request	None
osdp_RSTAT	0x67	Reader Status Report Request	None
osdp_OUT	0x68	Output Control Command	Output settings
osdp_LED	0x69	Reader Led Control Command	LED settings
osdp_BUZ	0x6A	Reader Buzzer Control Command	Buzzer settings
osdp_TEXT	0x6B	Text Output Command	Text settings
osdp_RMODE	0x6C	… removed …	
osdp_TDSET	0x6D	Time and Date Command	Time and Date
osdp_COMSET	0x6E	PD Communication Configuration Command	Com settings
osdp_DATA	0x6F	Data Transfer Command	Raw Data
osdp_XMIT	0x70	… removed …	
osdp_PROMPT	0x71	Set Automatic Reader Prompt Strings	Message string
osdp_SPE	0x72	… removed …	
osdp_BIOREAD	0x73	Scan and Send Biometric Data	Requested Return Format
osdp_BIOMATCH	0x74	Scan and Match Biometric Template	Biometric Template
osdp_KEYSET	0x75	Encryption Key Set Command	Encryption Key
osdp_CHLNG	0x76	Challenge and Secure Session Initialization Rq.	Challenge Data
osdp_SCRYPT	0x77	Server Cryptogram	Encryption Data
osdp_CONT	0x79	Continue Sending Multi-Part Message	None
osdp_MFG	0x80	Manufacturer Specific Command	Any
osdp_SCDONE	0xA0	… removed …	
osdp_XWR	0xA1	See appendix	Defined in Appendix E

Replies

Name	Value	Meaning	Data
osdp_ACK	0x40	Command accepted, nothing else to report	None
osdp_NAK	0x41	Command not processed	Reason for rejecting command
osdp_PDID	0x45	PD ID Report	Report data
osdp_PDCAP	0x46	PD Capabilities Report	Report data
osdp_LSTATR	0x48	Local Status Report	Report data
osdp_ISTATR	0x49	Input Status Report	Report data
osdp_OSTATR	0x4A	Output Status Report	Report data
osdp_RSTATR	0x4B	Reader Status Report	Report data
osdp_RAW	0x50	Reader Data – Raw bit image of card data	Card data
osdp_FMT	0x51	Reader Data – Formatted character stream	Card data
osdp_PRES	0x52	… removed …	
osdp_KPD	0x53	Keypad Data	Keypad data
osdp_COM	0x54	PD Communications Configuration Report	Comm data
osdp_SCREP	0x55	… removed …	
osdp_SPER	0x56	… removed …	
osdp_BIOREADR	0x57	Biometric Data	Biometric data
osdp_FPMATCHR	0x58	Biometric Match Result	Result
osdp_CCRYPT	0x76	Client's ID, Random Number, and Cryptogram	Encryption Data
osdp_RMACI	0x78	Initial R-MAC	Encryption Data
osdp_MFGREP	0x90	Manufacturer Specific Reply	Any
osdp_BUSY	0x79	PD is Busy reply	
osdp_XRD	0xB1	See appendix	Defined in Appendix E

Appendix B - Function Code Definitions List

Function Code 001 – Contact Status Monitoring

This function indicates the ability to monitor the status of a switch using a two-wire electrical connection between the PD and the switch. The on/off position of the switch indicates the state of an external device.

The PD may simply resolve all circuit states to an open/closed status, or it may implement supervision of the monitoring circuit. A supervised circuit is able to indicate circuit fault status in addition to open/closed status.

Compliance Levels:

01 – PD monitors and reports the state of the circuit without any supervision. The PD encodes the circuit status per its default interpretation of contact state to active/inactive status.
02 – Like 01, plus: The PD accepts configuration of the encoding of the open/closed circuit status to the reported active/inactive status. (User may configure each circuit as "normally closed" or "normally open".)
03 – Like 02, plus: PD supports supervised monitoring. The operating mode for each circuit is determined by configuration settings.
The End-Of-Line circuit parameters are defined by the manufacturer of the PD.
04 – Like 03, plus: the PD supports custom End-Of-Line settings within the Manufacturer's guidelines.

Function Code 002 – Output Control

This function provides a switched output, typically in the form of a relay. The Output has two states: active or inactive. The Control Panel (CP) can directly set the Output's state, or, if the PD supports timed operations, the CP can specify a time period for the activation of the Output.

Compliance Levels:

01 – The PD is able to activate and deactivate the Output per direct command from the CP.
02 – Like 01, plus: The PD is able to accept configuration of the Output driver to set the inactive state of the Output. The typical state of an inactive Output is the sate of the Output when no power is applied to the PD and the Output device (relay) is not energized. The inverted drive setting causes the PD to energize the Output during the inactive state and de-energize the Output during the active state.
This feature allows the support of "fail-safe/fail-secure" operating modes.

03 – Like 01, plus: The PD is able to accept timed commands to the Output. A timed command specifies the state of the Output for the specified duration.
04 – Like 02 and 03 – normal/inverted drive and timed operation.

Function Code 003 - Card Data Format

This capability indicates the form of the card data is presented to the Control Panel.

Compliance Levels:

01 – the PD sends card data to the CP as array of bits, not exceeding 1024 bits.
02 – the PD sends card data to the CP as array of BCD characters, not exceeding 256 characters.
03 – the PD can send card data to the CP as array of bits, or as an array of BCD characters.

Function Code 004 – LED Control

This capability indicates the presence of and type of LEDs.

Compliance Levels:

01 – the PD support on/off control only
02 – the PD supports timed commands
03 – like 02, plus bi-color LEDs
04 – like 02, plus tri-color LEDs

Function Code 005 – Audible Output

This capability indicates the presence of and type of an Audible Annunciator (buzzer or similar tone generator)

Compliance Levels:

01 – the PD support on/off control only
02 – the PD supports timed commands

Function Code 006 – Text Output

This capability indicates that the PD supports a text display emulating character-based display terminals.

Compliance Levels:

00 – The PD has no text display support
01 – The PD supports 1 row of 16 characters
02 – the PD supports 2 rows of 16 characters
03 – the PD supports 4 rows of 16 characters
04 - … tbd.

Function Code 007 – Time Keeping

This capability indicates that the type of date and time awareness or time keeping ability of the PD.

Compliance Levels:

00 – The PD does not support time/date functionality
01 – The PD understands time/date settings per Command osdp_TDSET
02 – The PD is able to locally update the time and date

Function Code 008 – Check Character Support

All PDs must be able to support the checksum mode. This capability indicates if the PD is capable of supporting CRC mode.

Compliance Levels:

00 – The PD does not support CRC-16, only checksum mode.
01 – The PD supports the 16-bit CRC-16 mode.

Function Code 009 – Communication Security

This capability indicates the extent to which the PD supports communication security as as defined in

Compliance Levels:

This field is a bit map of the supported encryption algorithms

0x01 – (Bit-0) AES128 support
0x02 – (Bit-1) to be defined

Number of:

This field is encoded to represent the key exchange capabilities

0x01 – (Bit-0) default AES128 key, as defined in APPENDIX C
0x02 – (Bit-1) to be defined

Function Code 010 – Receive BufferSize

This capability indicates the maximum size single message the PD can receive.

Compliance Levels:

This field is the LSB of the buffer size

Number of:

This field is the MSB of the buffer size

Function Code 011 – Largest Combined Message Size

This capability indicates the maximum size multi-part message which the PD can handle.

Compliance Levels:

This field is the LSB of the combined buffer size

Number of:

This field is the MSB of the combined buffer size

Function Code 12 – Smart Card Support

This capability indicates whether the PD supports the transparent mode used for communicating directly with a smart card.

Compliance Levels:

0 - PD does not support transparent reader mode

1 - PD does support transparent reader mode

Number of:

unused, send 0x00

APPENDIX C - CRC Definition

All devices must be able to support the simple checksum defined earlier in this document. The preferred implementation uses the more robust error checking technique offered by a 16-bit Cyclic Redundancy Check character.

There are several well-documented algorithms in the public domain. The implementation selected for this protocol is commonly referred to as CRC16-CCITT. It is bases on the polynomial of x**16 + x**12 + x**5 + x, or more commonly represented as 0x1021. The initial value of the CRC register is all ones. The data bytes are passed through the CRC register most significant bit first. The message is always augmented with 16 zero bits.

Unlike the data storage convention used in other portions of this protocol, the most significant byte of the CRC register is stored first, followed by the least significant byte. This change is made to take advantage of the property of this CRC algorithm where the final result of computing the CRC of the entire received message, including the received cc characters, the final cc register should be zero.

This CRC algorithm is thoroughly addressed in the public domain. The following references provide a source for additional information:

The first link to start with should be:

http://www.joegeluso.com/software/articles/ccitt.htm

This site has a great overview, as well a several test strings with expected results for verification of specific implantations.

The following link is an excellent source for theoretical and practical discussion of cc methods. An especially valuable section deals with table driven implementations.

http://www.repairfaq.org/filipg//LINK/F_LINK_IN.html -- primary link

http://www.repairfaq.org/filipg/LINK/F_cc33.html#CRCV_003 --- table method

The block of C source code below is provided as an example of the direct table algorithm:

```
typedef unsigned int uint16;

static uint16 nCrcTblValid = 0;     // preset: CRC Table not initialized
static uint16 cCrcTable[256];       // CRC table - working copy

// generate the table for POLY == 0x1012
static int fCrcTblInit( uint16 *pTbl )
{     int ii, jj;
      uint16 ww;

      for (ii = 0; ii < 256; ii++) {
            ww = (uint16)(ii << 8);
            for (jj = 0; jj < 8; jj++) {
                  if ( ww & 0x8000 ) {
                        ww = (ww << 1) ^ 0x1021;
                  } else {
                        ww = (ww << 1);
                  }
            }
            pTbl[ii] = ww;
      }
      return 1;
}

// table based CRC - this is the "direct table" mode -
uint16 fCrcBlk( uint08 *pData, uint16 nLength)
{     uint16 nCrc;
      int ii;
```

```
      if ( nCrcTblValid == 0 ) {
            nCrcTblValid = fCrcTblInit(&cCrcTable[0]);
      }
      for ( ii = 0, nCrc = 0x1D0F; ii < nLength; ii++ ) {
            nCrc = (nCrc<<8) ^ cCrcTable[ ((nCrc>>8) ^ pData[ii]) & 0xFF];
      }
      return nCrc;
}
```

Note that the CRC table uses 512 bytes (256 two-byte entries). Depending on limitations on system resources, some implementations may prefer to place a pre-built table into Read Only Memory. Use the fCrcTblInit() function to generate the 256 entries, then format the output and place into the form of an initialized array, such as:

```
const uint16 CrcTable[256] =
{
0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50A5, 0x60C6, 0x70E7,
0x8108, 0x9129, 0xA14A, 0xB16B, 0xC18C, 0xD1AD, 0xE1CE, 0xF1EF,
0x1231, 0x0210, 0x3273, 0x2252, 0x52B5, 0x4294, 0x72F7, 0x62D6,
0x9339, 0x8318, 0xB37B, 0xA35A, 0xD3BD, 0xC39C, 0xF3FF, 0xE3DE,
0x2462, 0x3443, 0x0420, 0x1401, 0x64E6, 0x74C7, 0x44A4, 0x5485,
0xA56A, 0xB54B, 0x8528, 0x9509, 0xE5EE, 0xF5CF, 0xC5AC, 0xD58D,
0x3653, 0x2672, 0x1611, 0x0630, 0x76D7, 0x66F6, 0x5695, 0x46B4,
0xB75B, 0xA77A, 0x9719, 0x8738, 0xF7DF, 0xE7FE, 0xD79D, 0xC7BC,
0x48C4, 0x58E5, 0x6886, 0x78A7, 0x0840, 0x1861, 0x2802, 0x3823,
0xC9CC, 0xD9ED, 0xE98E, 0xF9AF, 0x8948, 0x9969, 0xA90A, 0xB92B,
0x5AF5, 0x4AD4, 0x7AB7, 0x6A96, 0x1A71, 0x0A50, 0x3A33, 0x2A12,
0xDBFD, 0xCBDC, 0xFBBF, 0xEB9E, 0x9B79, 0x8B58, 0xBB3B, 0xAB1A,
0x6CA6, 0x7C87, 0x4CE4, 0x5CC5, 0x2C22, 0x3C03, 0x0C60, 0x1C41,
0xEDAE, 0xFD8F, 0xCDEC, 0xDDCD, 0xAD2A, 0xBD0B, 0x8D68, 0x9D49,
0x7E97, 0x6EB6, 0x5ED5, 0x4EF4, 0x3E13, 0x2E32, 0x1E51, 0x0E70,
0xFF9F, 0xEFBE, 0xDFDD, 0xCFFC, 0xBF1B, 0xAF3A, 0x9F59, 0x8F78,
0x9188, 0x81A9, 0xB1CA, 0xA1EB, 0xD10C, 0xC12D, 0xF14E, 0xE16F,
0x1080, 0x00A1, 0x30C2, 0x20E3, 0x5004, 0x4025, 0x7046, 0x6067,
0x83B9, 0x9398, 0xA3FB, 0xB3DA, 0xC33D, 0xD31C, 0xE37F, 0xF35E,
0x02B1, 0x1290, 0x22F3, 0x32D2, 0x4235, 0x5214, 0x6277, 0x7256,
0xB5EA, 0xA5CB, 0x95A8, 0x8589, 0xF56E, 0xE54F, 0xD52C, 0xC50D,
0x34E2, 0x24C3, 0x14A0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
0xA7DB, 0xB7FA, 0x8799, 0x97B8, 0xE75F, 0xF77E, 0xC71D, 0xD73C,
0x26D3, 0x36F2, 0x0691, 0x16B0, 0x6657, 0x7676, 0x4615, 0x5634,
0xD94C, 0xC96D, 0xF90E, 0xE92F, 0x99C8, 0x89E9, 0xB98A, 0xA9AB,
0x5844, 0x4865, 0x7806, 0x6827, 0x18C0, 0x08E1, 0x3882, 0x28A3,
0xCB7D, 0xDB5C, 0xEB3F, 0xFB1E, 0x8BF9, 0x9BD8, 0xABBB, 0xBB9A,
0x4A75, 0x5A54, 0x6A37, 0x7A16, 0x0AF1, 0x1AD0, 0x2AB3, 0x3A92,
0xFD2E, 0xED0F, 0xDD6C, 0xCD4D, 0xBDAA, 0xAD8B, 0x9DE8, 0x8DC9,
0x7C26, 0x6C07, 0x5C64, 0x4C45, 0x3CA2, 0x2C83, 0x1CE0, 0x0CC1,
0xEF1F, 0xFF3E, 0xCF5D, 0xDF7C, 0xAF9B, 0xBFBA, 0x8FD9, 0x9FF8,
0x6E17, 0x7E36, 0x4E55, 0x5E74, 0x2E93, 0x3EB2, 0x0ED1, 0x1EF0
};
```

Appendix D – Encryption

Commands

Encryption Key Set (osdp_KEYSET)

This command transfers an encryption key from the CP to a PD.

Command structure: 2-byte header followed by variable length data

Byte	Name	Meaning	Value
0	Key_Type	Encryption method to use with this key	0x01 – Secure Channel Base Key 0x02 – tbd
1	Length	Length of the key in bytes	any
2 - N	Data	Key data	any

Notes:

The following notes apply to Key_Type = 0x01:

Used with the "Secure Channel protocol 03" Encryption Mode to transfer the Secure Channel Base Key (SCBK):

The Length shall be 16, and the Data[] aray shall contain the 128-bit Secure Channel base key (SCBK).

This command shall be sent by the CP and accepted by the PD only while the connection is "secure". The "secure" connection in this context shall meant that either a) the current connection is encrypted and the session keys are based on the current SCBK, or b) that the connection is inherently secure via physical security, such as CP/PD are connected via simple short cable. The "inherently secure" connection shall be asserted to the CP and to the PD by setting the devices into a special installation setup mode. The devices should exit the setup mode automatically after a successful completion of this osdp_KEYSET command.

Reply: osdp_ACK, osdp_NAK

Challenge and Secure Session Initialization Request (osdp_CHLNG)

This command is the first in the Secure Channel Session Connection Sequence (SCS-CS). It delivers a random challenge to the PD and it requests the PD to initialize for the secure session.

Command structure: 8-byte random number as the "challenge"

Byte	Name	Meaning	Value
0 - 7	Random Number	Random number generated by the CP (RND.A)	any

Command structure: none

Reply: osdp_ACK, osdp_NAK, osdp_CCRYPT

Server's Random Number and Server Cryptogram (osdp_SCRYPT)

This command transfers a block of data used for encryption synchronization.

Command structure: 24-byte structure, 1 8-byte field and 1 16-byte field

Byte	Name	Meaning	Value
0 - 16	Cryptogram	16-byte Server Cryptogram array	any

Refer to the Server Cryptogram paragraph below.

Reply: osdp_ACK, osdp_NAK, osdp_RMAC_I

Replies

Client's ID and Client's Random Number (osdp_CCRYPT)

This reply sends a block of data used for encryption synchronization, sent in response to osdp_CHLNG command.

Command structure: 32-byte structure

Byte	Name	Meaning	Value
0 - 7	Client ID	client's Unique Identifier (cUID)	any
8 - 15	Random Number	PD’s random number generated, (RND.B)	any
16 - 31	Cryptogram	16-byte Client Cryptogram array	any

Client Cryptogram Packet and the Initial R-MAC (osdp_RMAC_I)

This command transfers a block of data used for encryption synchronization, send in response to osdp_SCRYPT.

Command structure: 16-byte structure

Byte	Name	Meaning	Value
0 - 151	MAC_I	16-byte MAC array – initial MAC value	any

Cryptogram: refer to the Client Cryptogram paragraph below.

R-MAC is the initial value for the rolling MAC that is used during the Secure Channel Session. It is computed by encrypting the Server Cryptogram received in osdp_SCRYPT using S-MAC1, then encrypting the result using S-MAC2.

Encryption Method: osdp-sc

"sc" stands for Secure Channel

This section defines a specific security extension for OSDP based on the work in GlobalPlatform Inc's Secure Channel Protocol 03, Card Specification 2.2, 2009 Amendment D version 1.1.

Messages following this rule set are identified by the SEC_BLK_TYPE values assigned in this Subsection.

SEC_BLK_TYPE Assignment

Name	Value	Meaning	Direction
SCS_11	0x11	Begin new Secure Connection Sequence	CP to PD
SCS_12	0x12	Secure Connection Sequence step 2	PD to CP
SCS_13	0x13	Secure Connection Sequence step 3	CP to PD
SCS_14	0x14	Secure Connection Sequence step 4	PD to CP
SCS_15	0x15	Secure Session msg. w. MAC, no Data Security	CP to PD
SCS_16	0x16	Secure Session msg. w. MAC, no Data Security	PD to CP
SCS_17	0x17	Secure Session msg. with MAC & Data Security	CP to PD
SCS_18	0x18	Secure Session msg. with MAC & Data Security	PD to CP

osdp-sc based communication security is established and maintained during a communication SESSION. Built on the AES algorithm using a set of 128-bit keys, osdp-sc provides device authentication, data content security, and message authentication during the course of a session.

Terms and Abbreviations

Server – a Control Panel (CP)
Client - a Peripheral Device (PD)
Session - a secure communication connection
cUID - client's unique identifier. This is an 8-byte number, unique for each PD. The cUID assigned by PD's manufacturer and is intended to be available both in visual (label) and electronic form via the ID report.
The first 8 bytes of an osdp_CAP Reply, Device Identification report is suitable for this purpose, and its use for the cUID is hereby recommended. (Vendor Code 1, Vendor Code 2, Vendor Code 3, Model Number, Version, Serial Number ls, Serial Number lm, Serial Number hm (Serial Number ms is not used).
AES - Advanced Encryption Standard. FIPS 197. The AES algorithm using a 128-bit key (AES128) is the base algorithm for the osdp-sc described in this Appendix C.
CBC - Cipher Block Chaining. See SP 800-38A, modes of operations
ICV - Initial Chaining Vector, used by CBC, see SP 800-38A
MK - Master Key
SCBK - Secure Channel Base Key
MAC - Message Authentication Code
S-ENC - Session Key for ensuring data confidentiality (message encryption)
S-MAC1 - Session Key for Message Authentication, key 1
S-MAC2 - Session Key for Message Authentication, key 2
C-MAC - Command MAC (for packets from CP to PD)
R-MAC - Reply MAC (for packets from PD to CP)

General Overview

The Master Key (MK) is the foundation of the osdp-sc data security environment. The Control Panel, acting as the Server, contains the MK. The PDs are only loaded with a Secure Channel Base Key (SCBK) unique to each PD. The SCBK is computed by applying each PD's cUID and the MK to a key diversification algorithm.

To establish a secure connection between a CP and a PD, the PD presents its cUID in plain text. The CP performs the key diversification on the cUID and computes the PD's SCBK, thus establishing the

common key for the secure session.

Once the SCBK has been shared between the CP and PD, a set of three separate keys are established for the remaining of the communication "session": S-ENC, S-MAC1, and S-MAC2.

Each communication packet shall contain an encrypted message block for data privacy using S-ENC and authenticated using S-MAC1 and S-MAC2 with non-repeating ICVs.

The Process

In order to establish a session using the secure channel protocol, the Client and the Server must be mutually authenticated with each other and in the same process, a set of keys are established for this session. The secure channel session is terminated and the session keys are destroyed whenever any error is detected in the secure channel protocol. For example, the secure channel session can be terminated by either party by forcing a timeout, or by simply sending an invalid MAC.

The following steps define the osdp-sc Secure Channel Session Connection Sequence (SCS-CS), and also define the SEC_BLK_TYPE values assigned to each SCS step:

Secure Channel Session Connection Sequence (SCS-CS)

The following four steps initialize a Secure Channel Session. For these four commands SEC_BLK_DATA[0] is used to select SCBK or SCBK-D for the connection sequence. SEC_BLK_DATA[0] is set to 1 to select SCBK, and is set to 0 to select SCBK-D. The message check mode is CRC during this sequence.

SCS_11 CP->PD

The CP sends SCS_11 code in SEC_BLK_TYPE to begin a new SCS-CS. The SEC_BLK_DATA[0] is used to select SCBK or SCBK-D for the connection sequence. SEC_BLK_DATA[0] is set to 1 to select SCBK, and is set to 0 to select SCBK-D. The message check method is crc16.

The CMND character is osdp_CHLNG with an 8-byte random number (RND.A[8]) as the server challenge.

If the PD does not support the security block, and/or specifically SCS_11, then the PD shall return the osdp_NAK response: error_code set to 0x05.

SCS_12 PD->CP

The PD responds with SCS_12 to acknowledge the beginning a new SCS-CS. The SEC_BLK_DATA[0] is used to select SCBK or SCBK-D for the connection sequence. SEC_BLK_DATA[0] is set to 1 to select SCBK, and is set to 0 to select SCBK-D. The message check method is crc16.

The PD performs the following operations:

a) generates its own 8-byte random number (RND.B[8]), and

b) generates a set of session keys: S-ENC, S-MAC1, and MAC2, using the server's random number, RND.A[8], along with SCBK – see "**Session Key Derivation**" paragraph below.

c) Generate the Client Cryptogram - see "**Client Cryptogram**" paragraph below.

The REPLY is osdp_CCRYPT, returning the PD's ID (cUID), its random number, and the Client Cryptogram.

SCS_13 CP->PD

The CP continues by sending SCS_13 code in SEC_BLK_TYPE. The SEC_BLK_DATA[0] is used to select SCBK or SCBK-D for the connection sequence. SEC_BLK_DATA[0] is set to 1 to select SCBK, and is set to 0 to select SCBK-D. The message check method is crc16.

After receiving the osdp_CCRYPT in the SCS_12 reply, the CP will generate:

a) the PD's SCBK using its MK and cIUD – see "**Key Diversification**" paragraph below

b) generates a set of session keys: S-ENC, S-MAC1, and MAC2, using the server's random number, RND.A[8], along with SCBK – see "**Session Key Derivation**" paragraph below.

c) the Server Cryptogram – see "**Server Cryptogram**" paragraph below

The CP then formats and sends CMND osdp_SCRYPT, posting the Server Cryptogram.

SCS_14 PD->CP

The PD responds with SCS_14. The SEC_BLK_DATA[0] is used to select SCBK or SCBK-D for the connection sequence. SEC_BLK_DATA[0] is set to 1 to select SCBK, and is set to 0 to select SCBK-D. The message check method is crc16.

The PD processes the osdp_SCRYPT command and verifies the Server Cryptogram:

a) if the Server Cryptogram is ok, then

1) sec_blk_data[0] is set to "0x01" indicating that the Server Cryptogram in SCS_13 was accepted, and

2) generates the Intial MAC reply (osdp_RMAC_I) – as defined for the osdp_RMAC_I reply.

b) else (the Server Cryptogram test failed), the

1) sec_blk_data[0] is set to "0xFF" indicating that the Server Cryptogram in SCS_13 was not accepted, and secure connection sequence cannot proceed. Both the CP and the PD must begin a new secure connection sequence with SCS_11 – possibly using SCBK-D.

2) the REPLY code is set to the osdp_NAK response, with the error_code set to 0x05.

Note: successful completion of the first four steps confirms that the SCBK is valid, and that both sides have the full complement of the keys derived for this session: S-ENC, S-MAC1, and S-MAC2. Also, the R-MAC is the initial ICV value that will be rolling throughout the session.

Communication during a Secure Channel Session

The successful completion of the synchronization sequence SCS_11 through SCS_14, is followed by continuing to exchange the following OSDP_SC packet types. The message check method applied to all packets with SEC_BLK_TYPE set to SCS_15, SCS_16, SCS_17, and SCS_18 is the first four bytes of the 16-byte MAC computed for the message, as defined in the Message Authentication Code Generation paragraph, below.

SCS_15 CP->PD

The data field is sent in plain text (unencrypted)

Note: this form provides Message Authentication, but no Data Security. Therefore, it is intended to be used ONLY with the osdp_POLL command in "released" code. Certain development and test modes may use this form for testing.

SCS_16 PD->CP

The data field is sent in plain text (unencrypted)

Note: this form provides Message Authentication, but no Data Security. Therefore, it is intended to be

used ONLY with the osdp_ACK Reply in "released" code. Certain development and test modes may use this form for for testing.

SCS_17 CP->PD:

Data of the command is padded and encrypted using S-ENC key

Note: this form shall be used with all other commands (non-Poll)

SCS_18 PD->CP

Data of the reply is padded and encrypted using S-ENC key

Note: this form shall be used with all other replies (non-ACK)

Note: SCS_15 and SCS_16 provide for an efficient POLL/ACK cycle while maintaining encryption sync and message integrity. The assumption is made that there is no significant loss of security by sending POLL/ACK messages in plain text. If this becomes an issue, a site dependent security setting could disable the use of SCS_15/SCS_16 support, forcing even POLL/ACK messages to use the SCS_17/SCS_18 form.

Algorithms and Support Functions

Key Diversification

The Secure Channel Base Key (SCBK) is the secret key between the CP and the PD that is used to establish cryptographic synchronization. To support site based key management where it is not feasible to distribute the PDs' SCBKs to the CPs, the following algorithm allows for computation of unique SCBKs for each PD based on the PD's cUID and a site specific Master Key:

SCBK = Enc(cUID || (~cUID), MK)

The above equation means that the concatenated 8-byte cUID and the one's complement inverse of the cUID are encrypted by applying the AES128 algorithm using MK as the key. The nature of the AES block transform algorithm guarantees that the resultant SCBKs are unique as long as the cUIDs are unique.

Session Key Derivation

A set of three keys are derived and used for each secure communication session. The derivation operation uses the SCBK and encrypts a data block generated for each key. The data block for each key is as follows:

S-ENC: 0x01,0x82,rnd[0],rnd[1],rnd[2],rnd[3],rnd[4],rnd[5],0,0,…

S-MAC1: 0x01,0x01,rnd[0],rnd[1],rnd[2],rnd[3],rnd[4],rnd[5],0,0,…

S-MAC2: 0x01,0x02,rnd[0],rnd[1],rnd[2],rnd[3],rnd[4],rnd[5],0,0,…

The data fields rnd[0] through rnd[5] are the first 6 bytes of RND.A[8], the 8-byte random number generated by the CP. RND.A[8] is transferred to the PD at the time the secure connection is being established.

Client Cryptogram

The Client Cryptogram is computed by encrypting the concatenated RND.A[8] and RND.B[8] using key S-ENC. RND.A[8] is generated by the CP (server) and RND.B[8] is generated by the PD (client).

ClientCryptogram = ENC(RND.A[8] || RND.B[8], S-ENC)

Server Cryptogram

The Server Cryptogram is computed by encrypting the concatenated RND.B[8] and RND.A[8] using key S-ENC. RND.A[8] is generated by the CP (server) and RND.B[8] is generated by the PD (client).

ServerCryptogram = ENC(RND.B[8] || RND.A[8], S-ENC)

Padding

When used, the padding shall be performed as follows:

Append the character 0x80 to the data block, then continue to append as many characters of 0x00 as are required to make the size of the data block to be evenly divisible by the block size of 16.

Message Authentication Code (MAC) Generation

General: MAC is computed for and appended only to messages whose SEC_BLK_TYPE is SCS_15, SCS_16, SCS_17, and SCS_18, and the AES algorithm is applied in CBC mode using S-MAC1 as the key for all blocks, except the last one, and using S-MAC2 as the key for the last block.

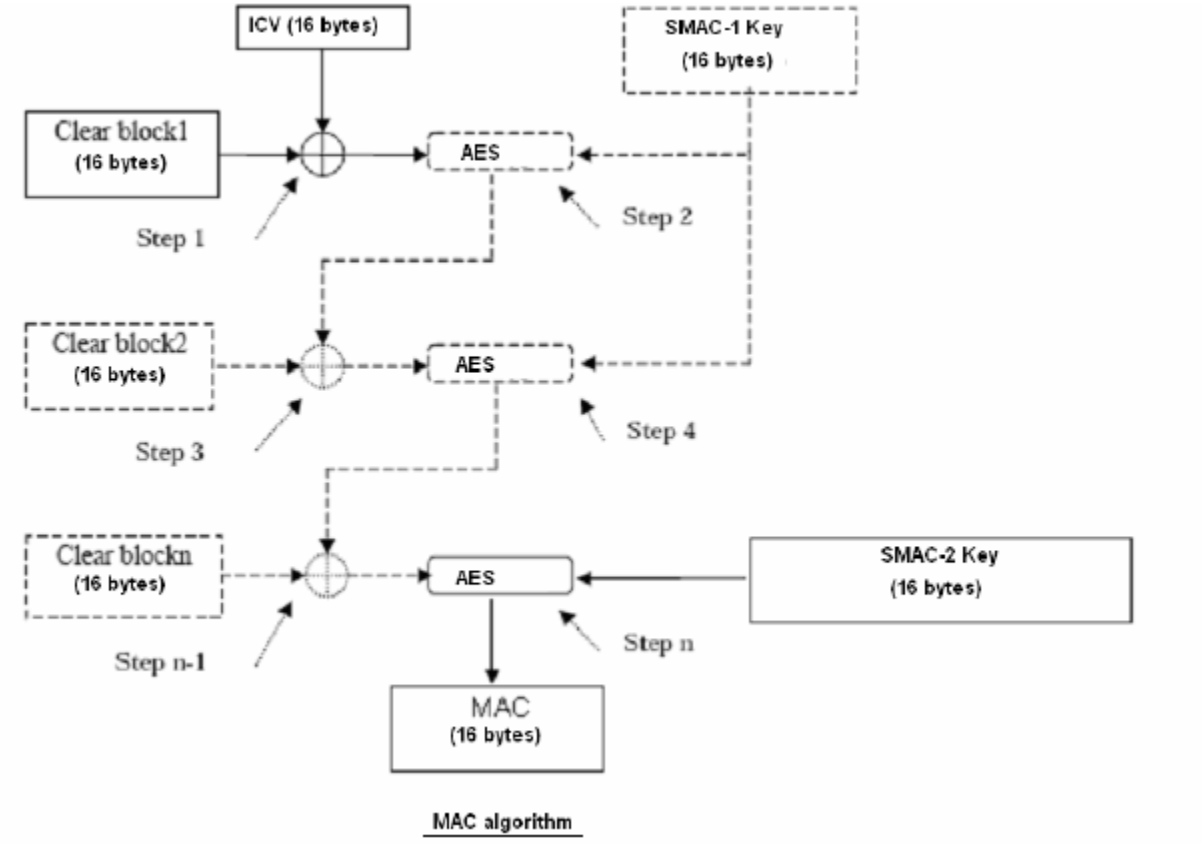
ICV values: The ICV is initialized during the Secure Connection Sequence by the PD and is passed to the CP during SCS_14 in reply osdp_RMAC_I.

R-MAC – the ICV value for generating the R-MAC is the previously received C-MAC.

C-MAC – the ICV value for generating the C-MAC is the previously received R-MAC.

After the initial osdp-sc setup, in order to reduce the message size and transmission time overhead, the messages will contain only a partial MAC. For messages whose SEC_BLK_TYPE is SCS_15, SCS_16, SCS_17, and SCS_18 the first four bytes of the computed MAC are actually sent. The MAC verification will locally generate the full MAC[16] and compare the actual bytes that were received.

The message portion that the MAC is applied to is the entire message including the Start of Message (53).



Note – If the message size is same as the block size (16 bytes) then only SMAC-2 key will be used as shown in the above diagram.

The Wrap Operation for Security Block Types SCS_15, SCS-16, SCS_17, and SCS_18

1. The message check data field appended to all packets whose SEC_BLK_TYPE is SCS_15, SCS_16, SCS_17, and SCS_18 is the first four bytes of the MAC. Note that as the table in the beginning of Section 3 shows, the MAC is replaces the checksum or the crc bytes.

2. Adjust the message length value to reflect the 4 byte MAC in the total length.

Note: Steps 3, 4, and 5 apply only to SCS_17 and SCS_18 ----->>>

3. "data" bytes of command or reply are padded to be of even multiple of the encryption block size.

4. Adjust the message length member of the header to reflect the padded length of the message.

5. Encrypt the Command/Reply Block using CBC mode with S-ENC key and ICV is the one's complement of the MAC of the last message received from the device the message is being prepared for.

Note: Steps 3, 4, and 5 apply only to SCS_17 and SCS_18 <<<-------

6. The message (header, sec block, command/reply, data block) is padded to form an even multiple of the encryption block length for the MAC calculation. (If the message size, prior to padding, is an even multiple of the block size, then no padding is performed.)

7. The MAC is computed by encrypting the padded message using CBC mode, ICV=previously received MAC, key=S-MAC1 for all blocks, except S-MAC2 is used for the last block. The result of the last CBC cycle is the MAC. The partial results of the CBC operation are discarded.

8. The first four of bytes of the MAC are placed at the end of the message (right after the encrypted data[] block) before the CRC/Checksum.

Note that the message length is adjusted to reflect the additional length resulting from the data Block padding, but it is NOT adjusted for the padding that was applied to the message for the MAC calculation since this padding is strictly used for the computation of the MAC and is NOT sent as part of the message.

The Unwrap Operation

1. The MAC is removed from the security block for later use.

2. If the message size (up to the first MAC position) is an even multiple of the block size, then no padding is performed, otherwise the message as padded as defined above. The MAC is then computed by encrypting the padded message using CBC mode, ICV=previously sent MAC, key=S-MAC1 for blocks, except S-MAC2 is used for the last block. The result of the last CBC cycle is the MAC. The partial results of the CBC operation are discarded.

3. Compare the first four bytes of the computed MAC to the four byte MAC received.

4. Decrypt the received Command/Reply Block using CBC mode, ICV=one's complement inverse of the last MAC sent, S-ENC as the key. Trim the pad from the decrypted message and adjust the command/reply data length.

5. Process the received command/reply message.

Field Deployment and Configuration:

To provide a means for the configuration of osdp-sc without HOST intervention:

Ideally, the installer should be able to link a CP and all the PDs that are intended to be connected to it via osdp-sc in a simple and efficient manner.

osdp-sc requires that key synchronization is established between the devices that are to operate with each other. This means that the CP, and only the CP, needs to know the Master Key (MK). A unique Secure Channel Base Keys (SCBK) is to be derived for and loaded into each PD with which the CP is to communicate.

Therefore, the following procedure is proposed, but not prescribed:

Establish and define a Default SCBK (SCBK-D) for use during installation. The SCBK-D shall be published in this document and will be supported by all implementations of osdp-sc. The SCBK-D will be used if the CP cannot create the PD's SCBK. This condition is detected as the key synchronization step has failed, and we have "install" option set - meaning that we are allowed to use the SCBK-D.

The following can be reader comm. setting options:

OSDP, no comm. security required

osdp-sc, "installation mode": auto sync using SCBK-D as needed

osdp-sc, full security (must use proper SCBK, no SCBK-D sync allowed)

The CP secure connection mode for the CP is controlled via the OSDP driver mode setting received from the HOST. The PD units could have an installer accessible selection (programming card, DIP switch, etc.) that, while enabled, would enable the SCBK-D based connection. The installer would disable this setting after the units have been synced. PDs that do not have installer selectable components may elect to allow the use of synchronization with the SCBK-D for xx seconds following power-up.

Appendix E – Extended Write Command and Extended Read Reply

Overview

The "extended write" command and the "extended read" reply support Peripheral Devices, such as smart card readers, which are capable of performing intelligent operations. The ability to handle complex tasks is referred to as "Behavior Profiles", or more simply "Profiles", throughout this Appendix. The capabilities and specific nature of the various Profiles are documented in subsections below.
(The messages in this section provide complete functional replacement for the following messages that were removed from the main body of the specification: osdp_XMIT, osdp_RMODE, osdp_SPE, osdp_SCDONE, osdp_SCREP, osdp_PRES, and osdp_SPER.)

Behavior Profiles

XRW_PROFILE	Behavior Profile Description
0x00	Default – no specific Behavior Profile is in effect
0x01	Transparent smart card interface support
0x02	- Unassigned

XRW_PROFILE == 0 – No Specific Behavior Profile is in use - osdp_XWR commands support the read back and the setting of the PD's profile.

XRW_PROFILE == 1 – This Behavior Profile supports transparent operations between the CP and a Smart Card. Note that the use of this profile may require licensing – refer to US Patents 6575360 & 7853785.

General Guidelines Regarding the Use of Behavior Profiles

The combination of **XRW_PROFILE** and **XWR_PCMND** are both required to uniquely identify the command, and the combination of **XRW_PROFILE** and **XRD_REPLY** are both required to uniquely identify the reply. Therefore for clarity, they should be handled and referenced as pairs.

This specification allows for the condition where a given PD may support several BPs, and it may even support those several BPs simultaneously. The guideline regarding multiple BPs is that the PD shall process all profile specific commands that it is capable of processing. If these commands create a conflict, then the PD shall return an error code in response.

Certain Behavior Profiles may be enabled to operate in the background. Scanning for card presence is an example of a background operation that will return an osdp_XRD reply to a poll command.

E.1 Extended Write Command (osdp_XWR)

Command Structure: 2 byte command structure, followed by an optional data block.

Extended Write Command Structure

Byte	Name	Meaning	Value
0	XRW_PROFILE	Extended READ/WRITE Profile, details below	Any
1	XWR_PCMND	XRW_PROFILE dependent command code	per profile spec
3 – n	XWR_PDATA	Optional - XWR_PCMND dependent data	per profile spec

Reply: osdp_ACK, osdp_NAK, osdp_XRD

E.1.1 Profile specific command codes for XRW_PROFILE == 0

XWR_PCMND	XRW_PROFILE == 0: Command Message Description
0x01	Request the PD to return the current XRW_PROFILE in effect
0x02	Set the PD to the specified XRW_PROFILE

E.1.1.1 Profile Setting Read Request (osdp_PR00REQ)

The Profile-00 read setting command is a request to the PD to return its current background behavior profile setting.

Command structure: 2 byte profile and command spec, no additional command data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x00
1	XRW_PCMND	Read Profile Setting	0x01

Reply: osdp_XRD[XRW_PROFILE=0x00,XRD_REPLY=0x01], or osdp_NAK

E.1.1.2 Profile Set Command (osdp_PR00SET)

The Profile-00 set command sets the background behavior profile.

Command structure: 2 byte profile and command spec, no additional command data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x00
1	XRW_PCMND	Read Profile Setting	0x01
2	Profile code	The XRW_PROFILE to set for background operation	valid XRW_PROFILE

Reply: osdp_ACK, or osdp_NAK

E.1.2 Profile specific command codes for XRW_PROFILE == 1

XWR_PCMND	Command Message Description
0x01	Pass the APDU embedded in this command to the specified reader (was osdp_XMIT)
0x02	Notifies the designated reader to terminate its connection to the Smart Card (was osdp_SCDONE)
0x03	Instructs the designated reader to perform “Secure PIN Entry”

XWR_PCMND	Command Message Description
	(was osdp_SPE)
0x04	- Unassigned

E.1.2.1 Transparent Content Send Request (osdp_PR01XMIT)

The Profile-01 XMIT command passes a data packet to the PD (reader) to pass on the Smart Card.

Command structure: 2 byte profile and command spec, 1 byte reader number, followed by a data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-01 command	0x01
1	XWR_PCMND	Transparent Content Send Command	0x01
2	Reader Number	0 == Reader-00	any
3-n	APDU	Valid APDU to send to the smart card	any

Reply: osdp_ACK, or osdp_NAK

E.1.2.2 Profile Set Command (osdp_PR01SCDONE)

The Profile-01 Smart Card Connection Done command instructs the PD (reader) to disconnect from the Smart Card.

Command structure: 2 byte profile and command spec, no additional command data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x01
1	XRW_PCMND	Smart Card Connection Done	0x02
2	Reader Number	0 == Reader-00	any

Reply: osdp_ACK, or osdp_NAK

E.1.2.3 Request Secure PIN Entry Command (osdp_PR01SPE)

The Profile-01 Secure PIN Entry command instructs the PD (reader) to perform a local Secure PIN Entry sequence with the Smart Card. It also includes an APDU for the Smart Card. When the reader receives this packet, it autonomously prompts the user for their PIN, inserts the PIN into the APDU and sends it to the smart card. The reader should restore the display to its previous state when done processing the user input.

While processing this message, the reader should not add any keys to the keypad buffer.

Command Structure: a 22-byte header followed by a variable-length APDU

Byte	Name	Meaning	Value

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x01
1	XRW_PCMND	Secure PIN Entry Request	0x03
2	Reader Number	0 == Reader-00	any
3	bTimeOut	timeout in seconds (00 means use default timeout)	
4	bTimeOut2	timeout in seconds after first key stroke	
5	bmFormatString	Formatting USB_CCID_PIN_FORMAT_xxx	
6	bmPINBlockString	Bits 7-4 - bit size of PIN length in APDU bits 3-0 - PIN block size in bytes after justification and formatting	
7	bmPINLengthFormat	bits 7-5 RFU, bit 4 set if system units are bytes clear if system units are bits. bits 3-0 PIN length position in system units	
8	wPINMaxExtraDigit MSB	XXYY, where XX is minimum PIN size in digits, YY is maximum	
9	wPINMaxExtraDigit LSB		
10	bEntryValidationCondition	Conditions under which PIN entry should be considered complete	
11	bNumberMessage	Number of verification messages to display for PIN	
12	wLangId MSB	Language for messages	
13	wLangId LSB		
14	bMsgIndex	Message index (should be 00)	
15	bTeoPrologue	(3 bytes) T=1 I-block prologue field to use (fill with 00)	
16			
17			
18	ulDataLength MSB	length of APDU to be sent to the smart card	
19			
20			
21	ulDataLength LSB		
22	abData	APDU data to be sent to the smart card	

Below is an example of how this data structure is built on the host:

```
PIN_VERIFY_STRUCTURE *pin_verify;

pin_verify = (PIN_VERIFY_STRUCTURE *)bSendBuffer;

/* PC/SC v2.02.05 Part 10 PIN verification data structure */
pin_verify -> bTimerOut = 0x00;
pin_verify -> bTimerOut2 = 0x00;
pin_verify -> bmFormatString = 0x82; /* ascii, left justified, 0 offset from Lc,
```

```
                                                system unit bytes */
        pin_verify -> bmPINBlockString = 0x08;
        pin_verify -> bmPINLengthFormat = 0x00;
        pin_verify -> wPINMaxExtraDigit = HOST_TO_CCID_16(0x0408); /* Min Max */
        pin_verify -> bEntryValidationCondition = 0x02;/* validation key pressed */
        pin_verify -> bNumberMessage = 0x01;
        pin_verify -> wLangId = HOST_TO_CCID_16(0x0904);
        pin_verify -> bMsgIndex = 0x00;
        pin_verify -> bTeoPrologue[0] = 0x00;
        pin_verify -> bTeoPrologue[1] = 0x00;
        pin_verify -> bTeoPrologue[2] = 0x00;
        /* pin_verify -> ulDataLength = 0x00; we don't know the size yet */

        /* APDU: 00 20 00 00 08 30 30 30 30 00 00 00 00 */
        offset = 0;
        pin_verify -> abData[offset++] = 0x00;   /* CLA */
        pin_verify -> abData[offset++] = 0x20;   /* INS: VERIFY */
        pin_verify -> abData[offset++] = 0x00;   /* P1 */
        pin_verify -> abData[offset++] = 0x80;   /* P2 */
        pin_verify -> abData[offset++] = 0x08;   /* Lc: 8 data bytes */
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> abData[offset++] = 0xff;
        pin_verify -> ulDataLength = HOST_TO_CCID_32(offset);      /* APDU size */

        send_len = sizeof(PIN_VERIFY_STRUCTURE) + offset -1;
        /* -1 because PIN_VERIFY_STRUCTURE contains the first byte of abData[] */
```

Reply: osdp_ACK, or osdp_NAK

E.2 Extended Read Reply (osdp_XRD)

Sent in response to an osdp_XWR command, or as a "poll response".

Reply Structure: 2 byte reply structure, followed by an optional data block.

Byte	Name	Meaning	Value
0	XRW_PROFILE	Extended READ/WRITE Profile, details below	any
1	XRD_PREPLY	XRW_PROFILE dependent reply code	per profile spec
3 – n	XRD_PDATA	Optional - XWR_PREPLY dependent data	per profile spec

Profile specific reply codes for XRW_PROFILE == 1

E.2.1 Profile specific reply codes for XRW_PROFILE == 0

XRD_REPLY	Reply Message Description
0x00	General error indication: PD was unable to process the command
0x01	Returns the current XRW_PROFILE in effect

E.2.1.1 Profile-00 NAK or Error reply (osdp_PR00ERROR)

This may be sent as a poll response, or in response to any Profile-00 command (osdp_XWR | XRD_PROFILE == 0 | XWR_PCMND == any) to return an error or negative acknowledge (NAK) condition.

Reply structure: 2 byte profile and command spec followed by a single byte error code.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 reply	0x00
1	XRD_REPLY	This is an Extended Read/Write operation related error report	0x00
2	Error code	Error code – to be defined	any

E.2.1.2 Profile Setting report (osdp_PR00REQR)

This reply is sent in response to osdp_XWR|XRD_PROFILE==0|XWR_PCMND==osdp_PR00REQ and it returns its current background behavior profile setting in response to the request.

Command structure: 2 byte profile and reply spec, plus a single byte profile code.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x00
1	XRD_REPLY	Profile Setting Report	0x01
2	Profile code	XRW_PROFILE in use for background operation	valid XRW_PROFILE

E.2.2 Profile specific reply codes for XRW_PROFILE == 1

XRD_REPLY	Command Message Description
0x00	General error indication: PD was unable to process the command
0x01	Notifies the CP that the reader has detected a Smart Card (was osdp_PRES)
0x02	Returning an APDU embedded in this from the specified reader (was osdp_SCREP)
0x03	Reports that the reader has completed a “Secure PIN Entry” sequence (was osdp_SPER)
0x04	- unassigned

E.2.2.1 Profile-01 NAK or Error reply (osdp_PR01ERROR)

This may be sent as a poll response, or in response to any Profile-01 command (osdp_XWR | XRD_PROFILE == 1 | XWR_PCMND == any) to return an error or negative acknowledge (NAK) condition.

Reply structure: 2 byte profile and command spec followed by a single byte error code.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-01 reply	0x01
1	XRD_REPLY	This is an Extended Read/Write operation related error report	0x00
2	Error code	Error code – to be defined	any

E.2.2.2 Card Present Notification reply (osdp_PR01PRES == 0x01)

This reply is sent in response to an osdp_POLL indicating that a smart card has been detected and is available to communicate. This reply is used by smart card readers set to operate in background Profile == 1.

Command structure: 2 byte profile and reply spec, 1 byte reader number, followed by a data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-01 command	0x01
1	XRD_REPLY	This is a "card present" reply	0x01
2	Reader Number	0 == Reader-00	any

E.2.2.3 Transparent card data reply (osdp_PR01SCREP == 0x02)

This reply is sent in response to an osdp_POLL reporting a data packet received from a smart card by a reader readers set to operate in background Profile == 1.

Command structure: 2 byte profile and command spec, a reader number, followed by an APDU structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-00 command	0x01
1	XRD_REPLY	Smart Card Connection Done	0x02
2	Reader Number	0 == Reader-00	any
3	Status	Results of requested command	See List
4-n	APDU	APDU data from the card	any

E.2.2.4 Secure PIN Entry Complete reply (osdp_PR01SPER == 0x03)

This reply is sent in response to an osdp_POLL indicating that a Secure Pin Entry sequence has completed. This reply is used by smart card readers set to operate in background Profile == 1.

Command structure: 2 byte profile and reply spec, 1 byte reader number, followed by a data structure.

Byte	Name	Meaning	Value
0	XRW_PROFILE	This is a Profile-01 command	0x01
1	XRD_REPLY	This is a “card present” reply	0x03
2	Reader Number	0 == Reader-00	any
3	Status	Results of the SPE sequence	See list
4	Tries	Number of attempts before card “locks” itself	any

Appendix F – Test Vectors

CRC (CCITT-1021)

Example 1:

537F0D00046E00802500006E38

6E38 is the CRC here (in Little endian format)

Example 2:

53000900046100C066

C066 is the CRC here (in Little endian format)

Checksum

Example 1:

537F0C00006E00802500000F

0F is the Checksum here

Example 2:

5300080000610044

44 is the Checksum here

Sample Secure Channel establishment session:

Sample Shared SCBK_D key = "303132333435363738393A3B3C3D3E3F"

CP generates a 8 byte random number : "B0B1B2B3B4B5B6B7"

osdp_CHLG (530013000D03110076B0B1B2B3B4B5B6B73177):

Inside the PD:

- Generate 8 bytes random number; this session generates "A0A1A2A3A4A5A6A7"
- Generate session keys
  - SMAC1 – 5e 86 c6 76 60 3b de e2 d8 be af e1 78 63 73 32
  - SMAC2 – 6f da 86 e8 57 77 7e 81 13 20 35 75 82 39 17 2e
  - ENC – bf 8d c2 a8 32 9a cb 8c 67 c6 d0 cd 9a 45 16 82
- Generate host cryptogram (keep in memory) : 26 d3 35 6e 07 76 2d 26 28 01 fc 8e 66 65 a8 91
- Generate card cryptogram : fd e5 d2 f4 28 ec 16 31 24 71 ea 3c 02 bd 77 96

Response to osdp_CHLG would be :
53802B000D0312007600068E0000000000 A0A1A2A3A4A5A6A7 FDE5D2F428EC16312471EA3C02BD7796 F81E

osdp_SCRYPT (53001B000E0313007726D3356E07762D262801FC8E6665A89140B4):

Inside the PD:

- Validates the cryptogram sent by host (26D3356E07762D262801FC8E6665A891) with the one stored in memory (see osdp_CHLG)
- Generates RMAC. RMAC is generated by encrypting the host cryptogram by SMAC-1, the result of this encryption is then encrypted using SMAC-2. For this sample session the value of RMAC would be "b2 a3 00 57 eb 98 ba 22 29 ec 1f 87 56 62 b5 24"

Response to osdp_SCRYPT would be:
53801B000E03140178 B2A30057EB98BA2229EC1F875662B524 6EEB

References

[1] Interoperability Specification for ICCs and Personal Computer Systems
Revision 2.02.08
April 2010
www.pcscworkgroup.com/specifications/files/pcsc10_v2.02.08.pdf

[2] ANSI INCITS 378-2004
Information technology - Finger Minutiae Format for Data Interchange

[3] pgm - Netpbm grayscale image format
http://netpbm.sourceforge.net/doc/pgm.html