SecureDongle X

Developer's Guide

Version 1.0

http://www.securemetric.com

Disclaimer

The information contained in this Software Development Kit is for general guidance on using SecureMetric products. While SecureMetric have made every attempt to ensure the information contained in this Software Development Kit is believed to be accurate and reliable, however SecureMetric is not responsible for any errors or omissions, nor for any infringement of patents or other rights of third parties resulting from its use. Any third party trademarks or trade names are the property of their respective owners.

All the Products, Software Samples, Tools, Utilities and Documents contained in this Software Development Kit are the Intellectual Property of SecureMetric. SecureMetric reserves the right to make changes, updates, amendment, at anytime and without notice.

Revision History:

Date	Version	Description
November 2008	1.0	1st Edition

Software Developer's Agreement

All Products of SecureMetric Technology Sdn. Bhd. (SecureMetric) including, but not limited to, evaluation copies, diskettes, CD-ROMs, hardware and documentation, and all future orders, are subject to the terms of this Agreement. If developers do not agree with the terms herein, please return the evaluation package to us, postage and insurance prepaid, within seven days of their receipt, and we will reimburse developers the cost of the Product, less freight and reasonable handling charges.

1. Allowable Use - Developers may merge and link the Software with other programs for the sole purpose of protecting those programs in accordance with the usage described in the Developer's Guide. Developers may make archival copies of the Software.

2. Prohibited Use - The Software or hardware or any other part of the Product may not be copied, reengineered, disassembled, decompiled, revised, enhanced or otherwise modified, except as specifically allowed in item 1. Developers may not reverse engineer the Software or any part of the product or attempt to discover the Software's source code. Developers may not use the magnetic or optical media included with the Product for the purposes of transferring or storing data that was not either an original part of the Product, or a SecureMetric provided enhancement or upgrade to the Product.

3. Warranty - SecureMetric warrants that the hardware and Software storage media are substantially free from significant defects of workmanship or materials for a time period of twelve (12) months from the date of delivery of the Product to developers.

4. Breach of Warranty - In the event of breach of this warranty, SecureMetric's sole obligation is to replace or repair, at the discretion of SecureMetric, any Product free of charge. Any replaced Product becomes the property of SecureMetric.
Warranty claims must be made in writing to SecureMetric during the warranty period and within fourteen (14) days after the observation of the defect. All warranty claims must be accompanied by evidence of the defect that is deemed satisfactory by SecureMetric. Any Products that developers return to SecureMetric, or a SecureMetric authorized distributor, must be sent with freight and insurance prepaid.
EXCEPT AS STATED ABOVE, THERE IS NO OTHER WARRANTY OR REPRESENTATION OF THE PRODUCT, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

5. Limitation of SecureMetric's Liability - SecureMetric's entire liability to developers or any other party for any cause whatsoever, whether in contract or in tort, including negligence, shall not exceed the price developers paid for the unit of the Product that caused the damages or are the subject of, or indirectly related to the cause of action. In no event shall SecureMetric be liable for any damages caused by developers failure to meet developer's obligations, nor for any loss of data, profit or savings, or any other consequential and incidental damages, even if SecureMetric has been advised of the possibility of damages, or for any claim by developers based on any third-party claim.

6. Termination - This Agreement shall terminate if developers fail to comply with the terms herein. Items 2, 3, 4 and 5 shall survive any termination of this Agreement.

CE Attestation of Conformity

The equipment complies with the principal protection requirement of the EMC Directive (Directive 89/336/EEC relating to electromagnetic compatibility) based on a voluntary test.

This attestation applies only to the particular sample of the product and its technical documentation provided for testing and certification. The detailed test results and all standards used as well as the operation mode are listed in

Test report No.: 70407310011
Test standards: EN 55022/1998 EN 55024/1998

After preparation of the necessary technical documentation as well as the conformity declaration the CE marking as shown below can be affixed on the equipment as stipulated in Article 10.1 of the Directive. Other relevant Directives have to be observed.

FCC certificate of approval

This Device is in conformance with Part 15 of the FCC Rules and Regulations for Information Technology Equipment.

USB

This equipment is USB based.

WEEE

Dispose in separate collection.

Table of Contents

Chapter 1. Brief Introduction

SecureDongle X is a traditional secure storage dongle that is extremely simple to implement and at extremely low cost. It does not offer many of the advanced software protection methods but it can be the perfect choice for mass software developers who want simple protection with low budget.

SecureDongle X features include:

- ✓ Each SecureDongle X has 2560 bytes of read/write memory. This is a much larger memory space than what is available in most dongle products. Developer can make use of the provided memory to work as an external table specially aimed for software licensing data.
- ✓ SecureDongle X is an HID device so there is no driver required for supported Linux or Windows platforms. As long as a USB thumb drive can work on the computer, so does SecureDongle X.
- ✓ Each SecureDongle X has a globally unique hardware ID which can be a useful unique identifier in software licensing control.
- ✓ Multiple SecureDongle Xs can work together on the same computer.
- ✓ SecureDongle X supports USB only.

Chapter 2. SecureDongle X Features

User ID (UID) and Hardware ID (HID)

Each SecureDongle X contains a User ID (UID) as well as the globally unique Hardware ID (HID). Both the UID and HID are defined as 32-bit DWORD. The UID and HID are identifiers for the SecureDongle X.

The default UID is “715400947”; it is generated from the seed “12345”. SDX can be used even without changing the default UID; however, do take note that the first thing a hacker usually tests is the default value of anything; UIDs included.

The UID is reset by inserting a character string or seed code into a tool that generates the UID. The seed code must have a length of 64 bytes or less. The UID is generated inside the SecureDongle X’s hardware. The creation of the UID is entirely dependent on knowing the seed code. If a hacker is able to determine your UID, have access to the UID generation program, and have a SecureDongle X – they will still not be able to recreate your UID because your UID can only be created with that specific seed code. The seed code must be kept secret. It is important to note that the value of the seed code cannot be determined from the UID.

SecureDongle X Driver

The SecureDongle X is a USB device that uses the driver native to the Windows and Linux operating systems. When the SecureDongle X is inserted into the USB port, the operating system will prompt you to install a new device. With Windows 98, you may need the Win98 installation CD-ROM. You will not need the source media for Windows Me/2000/XP – the driver will install automatically.

The operating system combines the UID and HID to identify the SDX and install its driver. When a SecureDongle X first plugged into the computer, you will need to go through the driver installation process. When you reset the UID, the computer will see it as a new device and you will have to go through the process again. Once the UID is set, the computer will record the new UID/HID combination and you do not need to reinstall the driver. Keep in mind that since each SecureDongle X has a unique HID, and the operating system looks at the combination of UID and HID, every new SecureDongle X inserted into the computer will require installation of its driver. However, once the driver is installed for a specific UID/ HID combination, the SecureDongle X can be removed and later on reinserted to the computer without reinstalling the driver. Please note that the driver installation process is triggered by the SecureDongle X being inserted into the computer. When you change the UID, please remove and reinsert the SecureDongle X to install the driver for the new UID/HID combination.

SecureDongle X is initially configured with UID=715400947. The computer operating system will see this “out of the box” dongles as device number “1”. You should reset the UID so that the operating system can see a unique UID/HID combination. The UID should certainly be reset before the SecureDongle X is re-shipped and the UID generation call should not be included in the package sent to the end users.

Chapter 3. SecureDongle X API

In this chapter, we will introduce the SecureDongle X (SDX) well-designed API functions with elaborations. You can find the API samples with source codes available inside the Software Development Kit's CD-ROM.

SDX_Find: Find SDX(s) attached to the computer

EXTERN_C int WINAPI SDX_Find();

Return Value	<0	Error code
	=0	No SDX is attached
	>0	The number of attached SDX(s)

SDX_Open: Open specified SecureDongle X

EXTERN_C int WINAPI SDX_Open(int mode, DWORD uid, DWORD* hid);

Input	mode	This parameter indicates the way to open the SDX: - mode > 0: open the SDX according to the UID. The mode value is the SDX number, for example: uid=12345678, mode=2, this means it will open the second SDX with UID 12345678 - mode <= 0: open the SDX according to the HID, and the *hid parameter cannot be 0 We defined a Constant: HID_MODE=-1
	uid(User ID)	You need to specify the SDX UID and this UID is generated with the SDX Init Tool
	hid(Hardware ID)	Open SDX with HID of *hid The SDX HID will be returned to *hid regardless of how the SDX was opened.
Return	>= 0	Success. The opened SDX handle is returned.
	< 0	Error code. Please refer to the Chapter 4: Error Codes

SDX_Close: Close specified SecureDongle X

EXTERN_C void WINAPI SDX_Close(int handle);

Input	SecureDongle X handle. It is the handle returned from SDX_Open
Return	Error code. Please refer to the Chapter 4: Error Codes

SDX_Read: Read SecureDongle X content

EXTERN_C int WINAPI SDX_Read(int handle, int block_index, char* buffer512);

Input	handle	SDX handle. It is the handle returned from SDX_Open
	block_index	Block index. Specify the block to read. The value range is 0-4
	buffer512	Read buffer. The buffer must be at least 512 bytes to accommodate the 512-byte block size.
Return	Error code. Please refer to Chapter 4: Error Codes	

SDX_Write: Write to SecureDongle X

EXTERN_C int WINAPI SDX_Write(int handle, int block_index, char* buffer512);

Input	handle	SDX handle. It is the handle returned from SDX_Open
	block_index	Block index. Specify the block to write. The value range is 0-4
	buffer512	Write buffer. The buffer must be at least 512 bytes to accommodate the 512-byte block size.
Return	Error code. Please refer to Chapter 4: Error Codes	

SDX_Transform: Data hashing function

EXTERN_C int WINAPI SDX_Transform(int handle, int len, char* buffer);

Input	handle	SDX handle. It is the handle returned from SDX_Open.
	len	Length of buffer to transform. Maximum 32 characters.
	buffer	Content of data that user want to transform. SDX will return the result of transform to *buffer. The only use for this function is to transform your data into a hash string; you can then use the hash string to compare with another string that passes thru the SDX_Transform function too. The resulting data is not revertible to the original data.
Return	Error code. Please refer to Chapter 4: Error Codes	

SDX_GetVersion: Get SecureDongle X hardware version

EXTERN_C int WINAPI SDX_GetVersion(int handle);

Input	handle	SDX handle. It is the handle returned from SDX_Open.
Return	< 0	Error code. Please refer to the Chapter 4: Error Codes
	> 0	Success. The hardware version is returned.

SDX_RSAEncrypt: Encrypt with RSA and write to SecureDongle X

EXTERN_C int WINAPI SDX_RSAEncrypt(int handle, int startIndex, char* bufferData, int* len, char* Key512);

Input	handle	SDX handle. It is the handle returned from SDX_Open.
	startIndex	Start index. Specify the start index to write cipher text into SDX. The value range is 0-2559.
	bufferData	Plaintext that will be encrypted.
	len	Length of Data buffer to encrypt. On success, SDX will write the length of cipher text to len.
	Key512	Decryption key. The key must 512 Byte. On success, SDX will write the decryption key to Key512.
Return	Error code. Please refer to Chapter 4: Error Codes	

SDX_RSADecrypt: Read SecureDongle X content and Decrypt with RSA

EXTERN_C int WINAPI SDX_RSADecrypt(int handle, int startIndex, char* bufferData, int* len, char* Key512);

Input	handle	SDX handle. It is the handle returned from SDX_Open.
	startIndex	Start index. Specify the start index to read cipher text in SDX. The value range is 0-2559.
	bufferData	If success, SDX will write bufferData with plaintext.
	len	Length of cipher text to decrypt. If success SDX will write the length of plaintext to len.
	Key512	Key that is used to decrypt. The key size must 512 Bytes.
Return	Error code. Please refer to Chapter 4: Error Codes	

Notes for RSA Functions:

- We do not use Blocks rule for startIndex parameter of SDX_RSAEncrypt and SDX_RSADecrypt.
- For example, to start writing data from Block #1, byte #4, you use:
- 512 (size of Block #0) + 4 (Block #1 byte #4) – 1 (Because Blocks start from 0) = 512+4-1 = 515.
- A new block is every 512 bytes.
- For example (512 * 2 =) 1024 is the start of Block 2. (512 * 4) = 2048 is the start of Block 4
- SDX_RSAEncrypt and SDX_RSADecrypt are only available on the SDX.dll and SDX.lib; other languages that use ActiveX Component (OCX) and Delphi's SDX.dcu are currently not supported.
- Since the RSA functions on SDX are using 2048-bit keys, the processing time is quite slow to guarantee maximum data security.

Chapter 4. Error Codes

Constants	Value	Description
SDXERR_SUCCESS	0	Success
SDXERR_NO_SUCH_DEVICE	0xA0100001	Specified SDX is not found (parameter error)
SDXERR_NOT_OPENED_DEVICE	0xA0100002	Need to call SDX_Open first to open the SDX, then call this function (operation error)
SDXERR_WRONG_UID	0xA0100003	Wrong UID (parameter error)
SDXERR_WRONG_INDEX	0xA0100004	Block index error (parameter error)
SDXERR_TOO_LONG_SEED	0xA0100005	Seed character string is longer than 64 bytes when calling GenUID (parameter error)
SDXERR_WRITE_PROTECT	0xA0100006	Tried to write to write-protected SDX (operation error)
SDXERR_WRONG_START_INDEX	0xA0100007	Start index error (parameter error)
SDXERR_INVALID_LEN	0xA0100008	Invalid length (parameter error)
SDXERR_TOO_LONG_ENCRYPTION_DATA	0xA0100009	Cipher text length is to long (cryptography error)
SDXERR_GENERATE_KEY	0xA010000A	Generate key error (cryptography error)
SDXERR_INVALID_KEY	0xA010000B	Invalid key (cryptography error)
SDXERR_FAILED_ENCRYPTION	0xA010000C	Failed encrypt string (cryptography error)
SDXERR_FAILED_WRITE_KEY	0xA010000D	Failed write key (cryptography error)
SDXERR_FAILED_DECRYPTION	0xA010000E	Failed Decrypt string (Cryptography error)
SDXERR_OPEN_DEVICE	0xA010000F	Open device error (Windows error)
SDXERR_READ_REPORT	0xA0100010	Read record error (Windows error)
SDXERR_WRITE_REPORT	0xA0100011	Write record error (Windows error)
SDXERR_SETUP_DI_GET_ DEVICE_INTERFACE_DETAIL	0xA0100012	Internal error (Windows error)
SDXERR_GET_ATTRIBUTES	0xA0100013	Internal error (Windows error)
SDXERR_GET_PREPARSED_DATA	0xA0100014	Internal error (Windows error)
SDXERR_GETCAPS	0xA0100015	Internal error (Windows error)
SDXERR_FREE_PREPARSED_DATA	0xA0100016	Internal error (Windows error)
SDXERR_FLUSH_QUEUE	0xA0100017	Internal error (Windows error)
SDXERR_SETUP_DI_CLASS_DEVS	0xA0100018	Internal error (Windows error)
SDXERR_GET_SERIAL	0xA0100019	Internal error (Windows error)
SDXERR_GET_PRODUCT_STRING	0xA010001A	Internal error (Windows error)
SDXERR_TOO_LONG_DEVICE_DETAIL	0xA010001B	Internal error
SDXERR_WRONG_REPORT_LENGTH	0xA0100020	Unknown device (hardware error)
SDXERR_VERIFY	0xA0100021	Verification error (hardware error)
SDXERR_UNKNOWN_ERROR	0xA010FFFF	Unknown error (hardware error)